Introduction to Image and Video Processing - Project 2

Pie de Boer (i6272959)

Maastricht University
Department of Advanced Computing Sciences
Data Science and Artificial Intelligence

CONTENTS

1 DISCLAIMER

The initial intend while writing this paper was to utilize one dataset for the eigenfaces exercise. However, during the experimentation phase with images of faces **(Global-Face Dataset)** [1] taken in drastically differing conditions (e.g lighting, alignment), results were of varying quality. Considering this, it seemed reasonable to also investigate the performance of our algorithms with a dataset of faces in more consistent conditions **(ATT OLC Dataset)** [2]. This was done in order to find out whether the quality of the results was explained by the initial dataset or the algorithms. By including both datasets and presenting all the results, the paper showcases the most realistic progression of the research.

2 HUMAN PERCEPTION

In our first experiment, we delve into the realm of human vision and viewing distance using hybrid images. Fortunately, understanding this concept is quite intuitive! Individuals with myopia have all experienced those first moments where they were not able to detect the edges of distant objects or surfaces adequately. For instance, reading text on a distant whiteboard becomes challenging. This comes down to the fact that further away from our viewpoint, we experience less detail and vice-versa. A study from 2017, investigated hybrid images, the one we will study in this experiment, where they were used to asses the severity level of myopia [3]. This serves as anecdotal evidence that these images do not only look interesting but can also serve a practical purpose. Now, let's address the questions at hand: *What exactly are hybrid images, and how can we construct them?*

2.1 Hybrid Images and Viewing Distance

Hybrid images are images that can be perceived differently based on varying viewing distances [4]. The most famous example comes from the original paper by Olivia, were a hybrid image was built using a *filtered* images of Einstein and Monroe.

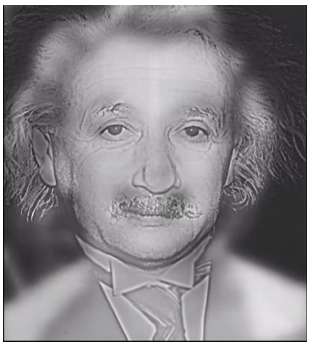

Fig. 1: Hybrid image of Monroe and Einstein [4]

Filtering, Viewing Distance and Hybrid Image Construction

Fusing *differently* filtered images will help use to gain fundamental understanding of human perception and viewing distance. For a pair of images in our dataset, we applied an ideal low pass filter (LPF) to the first image and an ideal high pass filter (HPF) to the second image. Applying a low-pass filter to an image, means high frequencies get damped. Vice-versa for a high-pass filter. These two processes respectively correspond to a loss of detail (LPF) or a sharpening of edges (HPF) [5].

After the images are filtered, we can fuse them in either the frequency or spatial domain. In the frequency domain, we add the 'frequency' representation of the filtered images. Therefore, many sinusoid (frequency components) with varying frequencies and amplitudes get added. Afterwards, using inverse Fourier it is brought back into spatial domain.

However, when the individual images are merged in the spatial domain, both filtered images need to be separately brought back into spatial domain using inverse Fourier transform. The spatial images, therefore, now correctly represent the filtered images in spatial domain. Due to linearity of Fourier transform [6], images added in both domains look (mostly) the same. It is possible that the slight difference observed in the fused images could be attributed to clipping, as there appears to be a slight variation in intensity. However, the psycho-visual effect remains unchanged.

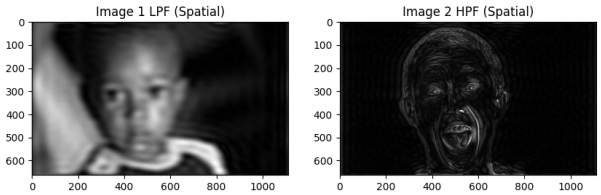

Fig. 2: The filtered images used for hybrid image construction

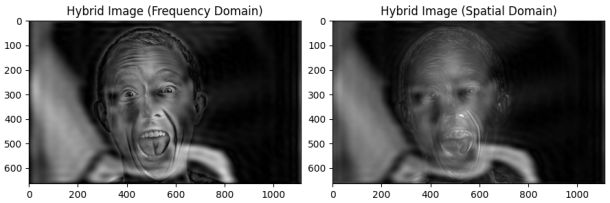

Fig. 3: Hybrid images obtained by merging filtered images in the spatial and frequency domains

Discussion

The obtained results consistent with the findings reported in Olivia's original paper [4]. Notably, we correctly perceive the young boy to be more pronounced on a further distance from our screen. This links to the fact that the low-pass filtered caused loss of the detail. Conversely, the image of the man becomes more apparent when the viewing distance decreases. By comparing our results with the hybrid image of Einstein/Monroe, we gained additional confidence that we correctly constructed a hybrid image.

3 Discrete Cosine Transform (DCT) Domain Watermarking

3.1 Watermark Insertion

This exercise investigates the usage of image watermarking. This technique can be utilized in a variety of applications related to authentication, copyright and security of digital media [7]. Therefore, it could be also used as measure against software piracy [8]. In our case, the process was conducted using grayscale images. However, the same logic applies to colored images as well. To implement this principle, the initial step involves adding a watermark to a specific image of interest. This can be achieved by incorporating noise from a distribution of interest (e.g. Gaussian) in the DCT-domain [5].

3.1.1 2D DCT of Image

In a block-wise manner, using block-size (8x8) the image of Lena, with dimensions (512x512), was brought into the DCT-domain. Figure 4, shows a spatial and DCT-domain block at pixel position 128.

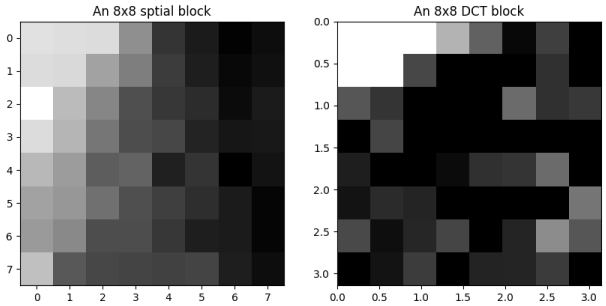

Fig. 4: Spatial and DCT domain block of lena image, $position = 128$

Additional, the full image DCT domain construction using (8x8) blocks is shown in figure 5.

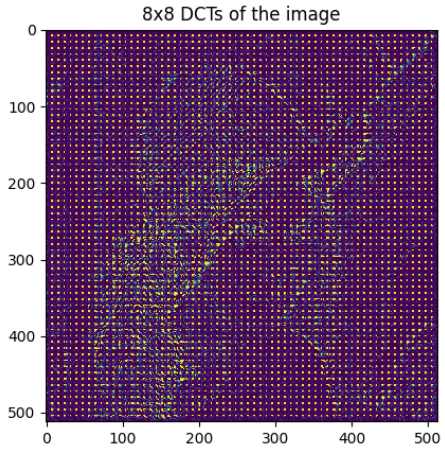

Fig. 5: 8x8 DCTs of Lena image

3.1.2 Reconstruction of Image

During the next step in our process, the image was reconstructed using the k most important *(non-DC)* coefficients. This process was done in a block-wise manner. It has to be noted, that excluding the DC coefficient had a significant influence on results. Therefore, the influence of the value k in presence of the DC-coefficient was investigated first. Subsequently, the reconstruction that exhibited non-observable degradation was chosen as a starting point for retaining only the AC components. Fortunately, the procedure using matrices as mask is intuitive, since the k most important coefficients follow the *zig-zag* pattern and the DC-coefficient is located at $(0, 0)$ [9]. A concise summary of this concept can be found in Figure 6.

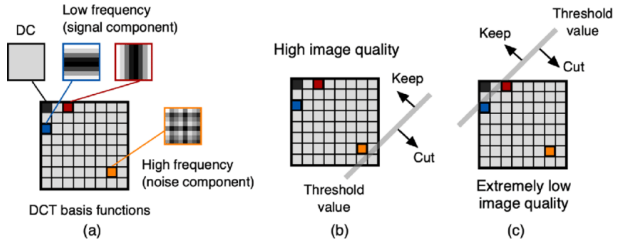

Fig. 6: The relationship between image quality and the value for k [10]

The impact of the k value is depicted in the following figures: Figure 7 and Figure 8. In order to facilitate the observation of the effect and consequently determine the optimal value of k more easily, the reconstructions include the DC component.

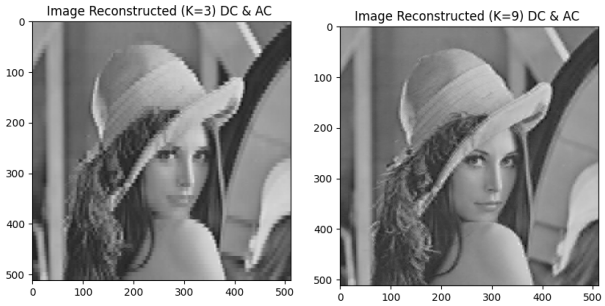

Fig. 7: Reconstructed images for $k = 3$ and $k = 9$ including both the DC and AC components.

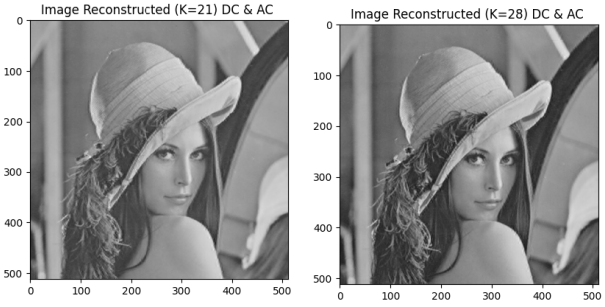

Fig. 8: Reconstructed images for $k = 21$ and $k = 28$ including both the DC and AC components.

Based on the visual inspection of the images, we decided to pick the kernel with $k = 27$. The value of 0 at the origin, showcases, that only the *non-DC* coefficients are taken into consideration.

$$\begin{vmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{vmatrix}$$

Fig. 9: Kernel for $k = 27$ that keeps the most important *non-DC* coefficients

In figure 10 we can see the reconstruction using only AC components with $k = 27$ and respectively, for the second image, all components.

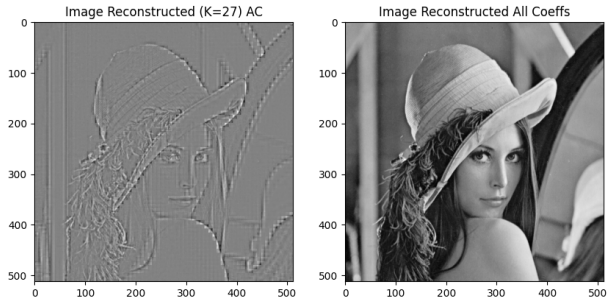

Fig. 10: The first image utilizing only AC components with $k = 27$, and the second image utilizing all k coefficients.

3.1.3 *Embedding the Watermark*

The objective is to add the watermark in such a way that the 9visual) quality is not degraded too much. This implies that the watermark is hidden. Finding the right strength, denoted α, for the watermark embedding was done through visual inspection. The noise originated from a Gaussian distribution with $\mu = 0$ and $\sigma^2 = 1$. Figures 11, 12, and 13 demonstrate the visual impact of increasing the value of $alpha$, enabling us to select the appropriate noise strength. This was done, while keeping the characteristics of the Gaussian noise consistent ($\mu = 0$, $\sigma^2 = 1$).

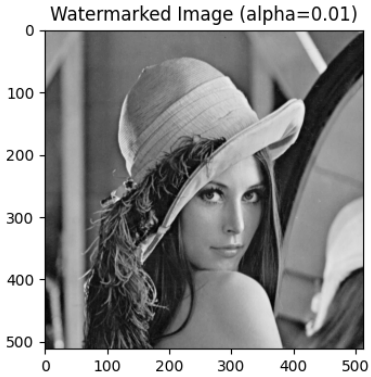

Fig. 11: Watermarked image with $\alpha = 0.01$, $\mu = 0$, and $\sigma^2 = 1$.

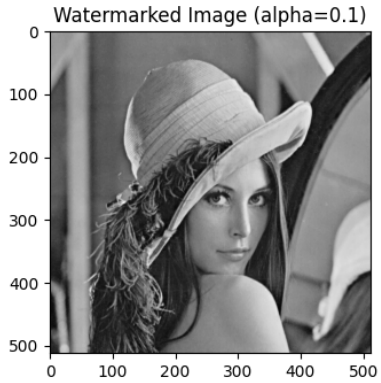

Fig. 12: Watermarked image with $\alpha = 0.1$, $\mu = 0$, and $\sigma^2 = 1$.

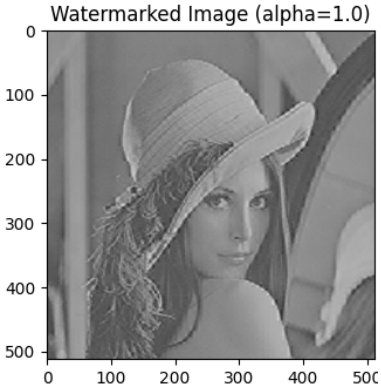

Fig. 13: Watermarked image with $\alpha = 1$, $\mu = 0$, and $\sigma^2 = 1$.

Based on the given images, $\alpha = 0.1$ was selected as an appropriate value for the watermark. It strikes a balance between hiding the watermark and avoiding an overly small α value.

Embedding of the watermark into the coefficients, was done according to the formula in Figure 14.

$$c_i' = c_i \cdot (1 + c\omega_i), 1 < i \leq K$$

Fig. 14: Formula for modifying DCT coefficients

3.1.4 The Difference Image

In order to gather the difference images, two approaches were used. One strategy, relied on computing the difference for each $8x8$ block in the DCT domain. Here, the watermarked and original image it's difference was computed by using a sliding window. The second strategy was to use the original and (re)constructed watermarked images, and subtract them in the spatial domain.

No plot is shown for the correctly constructed watermarked image, as it should be visually identical to the original image.

Based on the theory illustrated in Figure 15, selecting a low value for k corresponds to embedding the watermark into the low frequencies and therefore less detailed parts of the image.

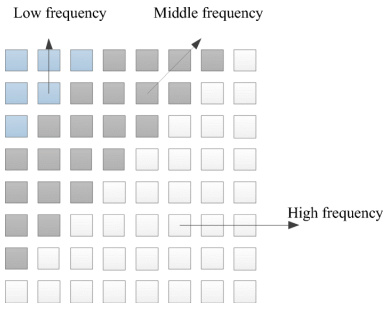

Fig. 15: DCT blocks representing low and high frequencies, corresponding to smooth areas and edges respectively [11].

This is most easily understood with some visual examples and the idea of the difference image. Since we can modulate the value for k, we should observe more 'blurry' difference images for lower values of k and more articulate/detailed images for higher values of k. A comparison between Figure 16 and Figure 17 clearly demonstrates the difference in sharpness and therefore the relation with the role of the value of k in the embedding process. The first figure 16 appears more blurry, while the second figure 17, with a higher value of k, exhibits more pronounced edges, particularly noticeable in the area around the eyes and the shape of hat.

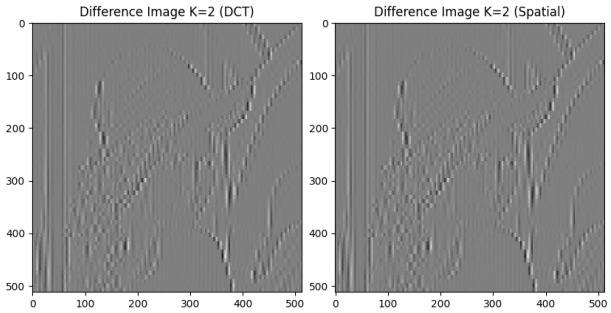

Fig. 16: DCT-domain and spatial difference images for *(non-DC)* $k = 2$

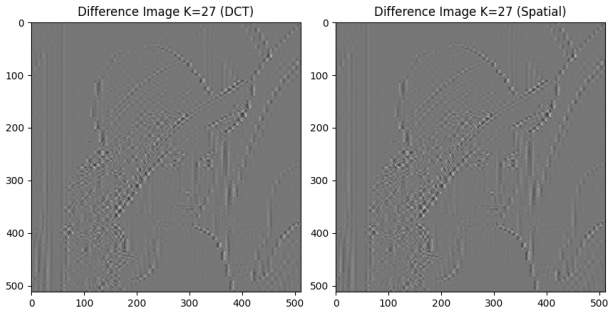

Fig. 17: DCT-domain and spatial difference images for *(non-DC)* $k = 27$

Finally, we examined the histogram of the watermarked image with $k = 27$ and Gaussian noise parameters $\mu = 0$ and $\sigma^2 = 1$. Observe the horizontal and vertical scales. Since a significant portion of the spectrum was removed after subtracting the watermark, it serves as a clear indication that the watermark was successfully concealed in the image. Additionally, the histograms of the watermarked image and the original are almost identical, providing further evidence of successful watermarking.

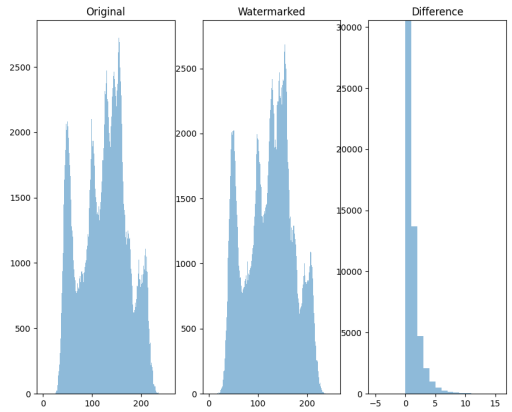

Fig. 18: Histogram comparison between the original, watermarked and difference images ($k = 27$, $\mu = 0$, $\sigma^2 = 1$)

3.2 Watermark Detection

In addition to inserting the watermark, it is of equal importance to accurately detect it. In a real-world context, this could mean verifying the authenticity of a multimedia object.

We start by making two watermarked images with the following attributes:

- **Watermarked Image 1**: non-DC coefficients $k = 27$, noise strength $\alpha = 0.1$, and Gaussian Noise parameters $\mu = 0$ and $\sigma^2 = 1$.
- **Watermarked Image 2**: non-DC coefficients $k = 27$, noise strength $\alpha = 0.1$, and Gaussian Noise parameters $\mu = 0$ and $\sigma^2 = 0.05$.

We could ask ourselves the question, if we could still successfully detect the watermark, in the case our Gaussian noise has a very small variance was utilized.

Since a successfully embedded watermark, is not visible to the human eye, the two images of the 'mystery images' are left out here as well as the DCT-block constructions.

Things become more interesting once we start approximating the watermark for our mystery images, this was done via the following formula 19.

$$\hat{\omega}_i = \frac{\hat{c}_i - c_i}{\alpha c_i}, \quad 1 \leq i \leq K$$

Fig. 19: Formula used to approximate watermark

Obviously, we want to know how good our approximation for the watermark is, therefore we look into the similarity between the approximated watermark $\hat{\omega}_i$ and the actual watermark ω_i. The similarity is investigated by calculating the correlation coefficient, for which the formulate used is 20.

$$\gamma = \frac{\sum_{i=1}^{K}(\hat{\omega}_i - \overline{\hat{\omega}})(\omega_i - \overline{\omega})}{\sqrt{\sum_{i=1}^{K}(\hat{\omega}_i - \overline{\omega})^2 \sum_{i=1}^{K}(\omega_i - \overline{\omega})^2}}, \quad 1 \leq i \leq K$$

Fig. 20: Formula to calculate correlation coefficient between original and approximated watermarks

Plotting the histograms for both the approximated and original watermark, can give us further guidance in whether we are correctly estimating the watermarks for both images. We can see that when the variance is very low, the histogram looks 'splashed' and therefore (most likely) wrong. A small variance, should only correspond to a tight region. If we correctly, calculated the similarity γ, this should mean that the value for it would be very low. Therefore, we can also directly derive how gamma behaves, when the watermark that is approximated is less similar. The value for γ goes down. In engineering terms, it would mean that we need to pick our variance σ^2 for the noise distribution of our watermark sufficiently high to be detected by this detection procedure, but at the same time, low enough (together with our value for α) to not corrupt the multimedia object.

The obtained results revealed that the first watermarked image had a value of $\gamma = 0.947$, while the second watermarked image had $\gamma = 0.117$. A threshold of approximately 0.90 could be considered as a starting point for correctly identifying the presence of the original media. However, further systematic testing involving different images distributions, and parameter settings would be necessary to arrive at a definitive conclusion. Any potential commercial implementation should only be considered after thorough investigation and evaluation.

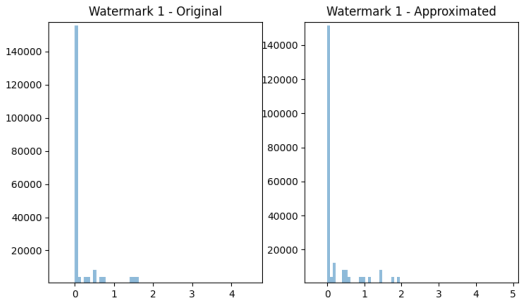

Fig. 21: Histogram of Watermarked Image 1 and Approximated Watermark ($\mu = 0$, $\sigma^2 = 1$)

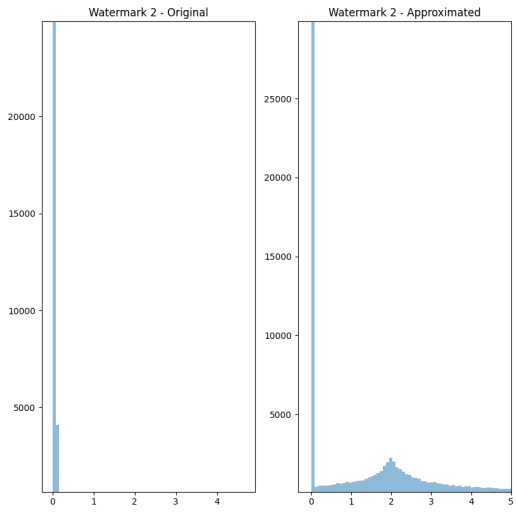

Fig. 22: Histogram of Watermarked Image 2 and Approximated Watermark ($\mu = 0$, $\sigma^2 = 0.01$), Notice the Scale of the Y-axis

4 Principle Component Analysis (PCA) - Recognition

This section investigates a relatively old approach from 1991, namely eigenfaces. The aim of this approach is to allow identification and detection of human-faces based on a set of eigenvectors [12]. According to the paper, due to similarity between faces, most variance, therefore similarity between faces, can be captured in a way lower dimension than where dataset lives. Linear combinations of the eigenvectors, allow us with dimensional reduced vectors, to reconstruct the face images [12]. This helps us in achieving drastic data reduction.

An helpful concept, is to find out how much principle components are needed to represent 95% of the variance. This is especially interesting to see for two data-sets, where one has relatively 'samey' images and the second has a lot more variation. It is fascinating to see, that indeed, less components are needed to capture 95% of the variance, when the images are less diverse. See figure 24 and figure 26.

In this paper, no further face detection using eigenfaces is investigated.

Datasets

The first dataset we constructed for the experiments, consisted of 100 images taken from **WorldFace Dataset** [1]. In total 30 images were picked from Caucasian persons, and 70 of African. Therefore, this allowed us to generate variation in the dataset. This gave an interesting results, for example for the mean face of the full dataset, see figure 28.

The second dataset consisted of images from ATT Laboratories **OCL** dataset [2]. Again, 100 images were picked. In order to make a more varied and interesting dataset, the first 7 subjects had no glasses, and the last 3 subjects did wore glasses. Both data sets and the explained variance plots are displayed below. Only 'all' eigenfaces are displayed for dataset world, but the concept is the same for both.

Fig. 23: 100 Images of faces taken from dataset (World) [1]

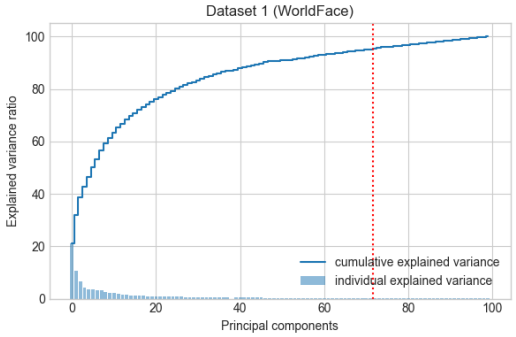

Fig. 24: PCA Explained Variance for dataset 1 (World) [1]

Fig. 25: 100 Images of faces taken from dataset (OCL)[2]

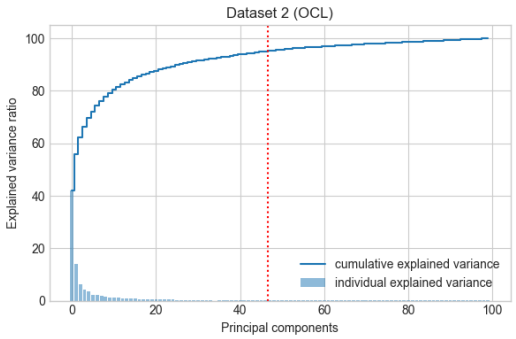

Fig. 26: PCA Explained Variance for dataset 2 [2]

For both datasets, in order to do PCA and further operations, they were pre-processed using the following strategy.

- Traverse the directory containing all images in the dataset
- Resize each image using bicubic interpolation to a dimension of (100, 100)
- Construct a matrix M with a size of (100, 100000)

The PCA algorithm consisted of the following high-level steps. In our source code, it is implemented in a *step-by-step* fashion.

Algorithm 1 Principal Component Analysis (PCA)

1: **Input:** Matrix M (dataset)
2: **Output:** Eigenfaces
3: **Step 1:** Compute the mean of each column for matrix M
4: **Step 2:** Subtract the mean from each column
5: **Step 3:** Compute the covariance matrix of the mean-centered matrix M
6: **Step 4:** Perform eigenvalue decomposition on the mean-centered matrix M
7: **Step 5:** Sort eigenvectors by absolute value of the eigenvalues
8: **Step 6:** Acquire eigenfaces by projecting top N eigenvectors onto the mean-centered matrix M

4.1 Finding the Eigenfaces

By doing the full PCA process manually for our first dataset, we could gather a excellent looking **10** first principle components/eigenfaces. The principle components are the eigenvectors of the covariance matrix sorted from biggest to lowest eigenvalue, giving a hierarchy of the most important ones [13]. Therefore, they show where most variance is captured with regards to the face (shapes). Thus, these are the eigenfaces and therefore correspond to the most prominent facial features.

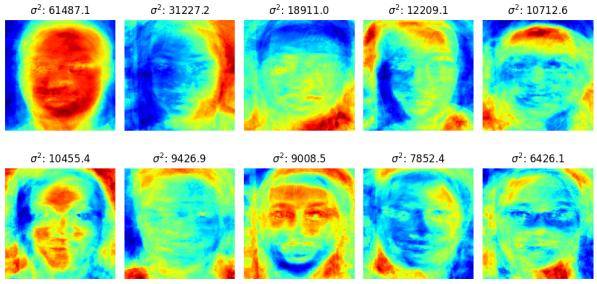

Fig. 27: The 10 Principle Components of the dataset **(World)** and their variance in JET colour map

The first principle component clearly corresponds to the shape of the head, which obviously is (mostly) the same for each subject. The second component shows that lighting and it's source position have a large influence. Continuing this train of thoughts, we could for example see in the fifth image, the shape of the top of the head 'roundness'. The sixth images, shows the form of the forehead. The eight image, demonstrates the vertical tilt, thus the way the chin comes forward. Investigating these pictures this way can tell us a lot! The unbalance of the dataset does not necessarily come forward in this inspection, since the main facial features of all 'people' are clearly the same.

In the mean-face, figure 28 however, we can observe that indeed it resembles more the features of the faces in the over-represented group. Strong cheekbones and a relatively accentuated forehead.

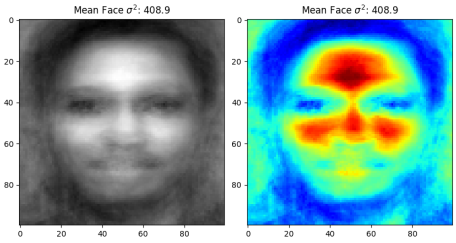

Fig. 28: Mean Face from the dataset 1 (World)

Furthermore, we incorporated the complete set of eigenfaces for this dataset. After comprehending the explanations provided earlier, you should now have a better understanding of the meaning of these images.

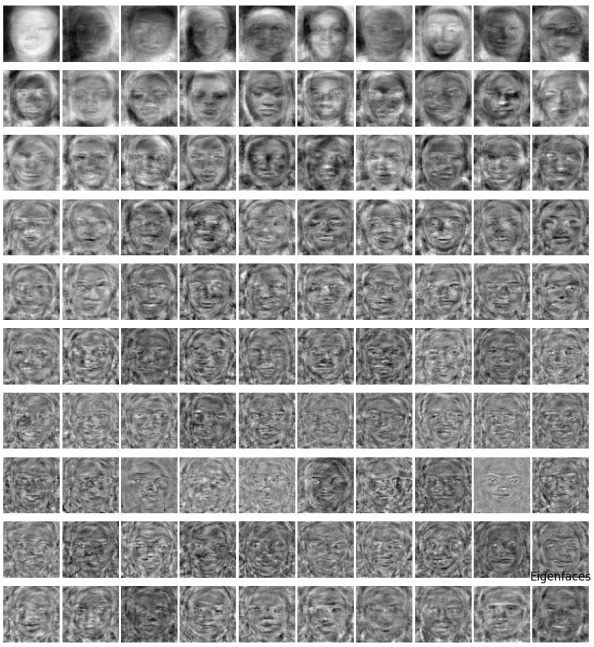

Fig. 29: All eigenfaces of the dataset 1 (World)

4.2 Reconstruction Using Eigenfaces

As mentioned before, two datasets were investigated, with second dataset (OCL) containing more same-ish looking images. It did help understanding face reconstruction a lot, since the faces were aligned, and mostly in the same lightning conditions.

Using less or more eigenfaces, means we capture more or less 'variance' which embeds features. Especially, the first principle components, capture the most variance and therefore most prominent feature. An obvious example would be, 'all faces are round'. As explained before.

The next step involves reconstructing faces, based on a varying amount of eigenfaces (and modified weights) and see how well they capture the faces. A new face is reconstructed according to the following formula [14]:

$$F = F_m + \sum_{i=1}^{n} (\alpha_i \cdot F_i)$$

Fig. 30: Formula for reconstructing new faces based on eigenfaces and weights

- F represents the new face.
- F_m represents the mean face (for the specific subject).
- F_i denotes an eigenface.
- α_i represents the corresponding weights.

The formula 30 mentioned before, demonstrated the reconstruction of a face using different numbers of eigenfaces and (modified) weights. The key question is how the variance is affected by the inclusion or exclusion of less/more eigenfaces. In theory, increasing the number of eigenfaces should lead to an increase in variance as more changes/features are captured. Conversely, limiting the range of weights is likely to decrease the variance and accentuate certain features.

4.3 Weights, Variance and Reconstructions

In our implementation, the modified weights existed of an array of equal size as the original weight. The modified weights were random integers in range $2 to 3$. Since the weights, have less 'varying' values, it helps accentuating certain features more, therefore logically capturing less variance, accentuating features. Using more faces to reconstruct the images, captures more details, therefore more variance, our results correspond with this (see results).

4.4 Results and Discussion

Description

- Faces Reconstructed:
  1) Data set (World) – Person 3 (Caucasian)
  2) Data set (World) – Person 5 (African)
  3) Data set (OCL) – Person 4 (no-glasses)
  4) Data set (OCL) – Person 10 (glasses)
- Mean faces were constructed using a subset of 2 or all eigenfaces.
- Faces were reconstructed using a subset of 2 or all eigenfaces, with both correct and modified weights.

For each figure (excluding mean faces), we provided information about the number of eigenfaces used, whether the weights were modified, and the corresponding variance in the title

Data set (World) – Person 3

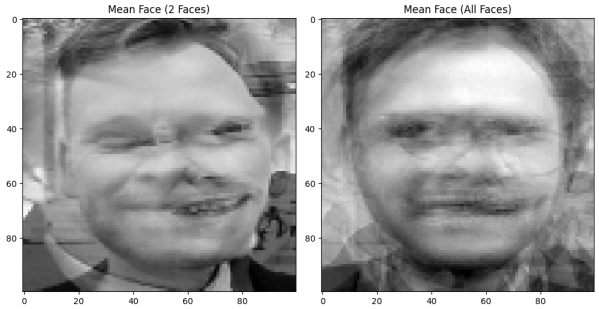

Fig. 31: Mean faces: 2 and $10(all)$ eigenfaces

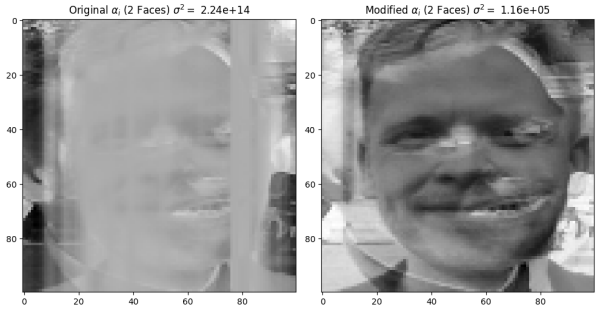

Fig. 32: Reconstructed faces: 2 eigenfaces (original-, modified weights)

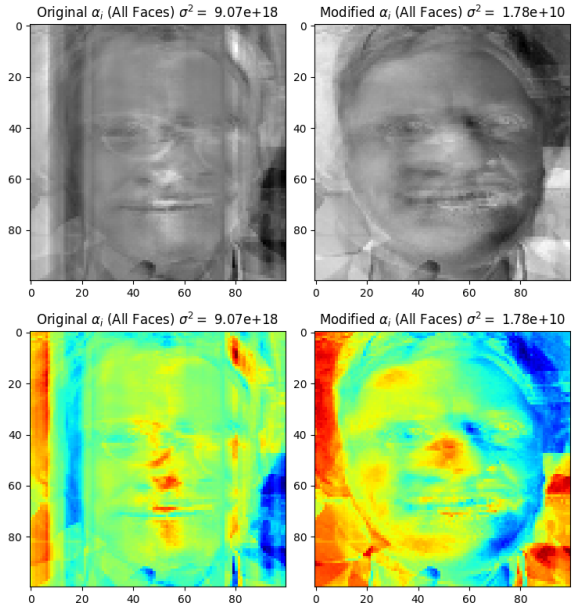

Fig. 33: Reconstructed faces: 10 eigenfaces (original-, modified weights)

Data set (World) – Person 5

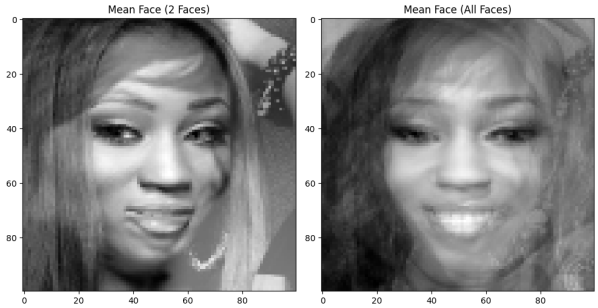

Fig. 34: Mean faces: 2 and $10(all)$ eigenfaces

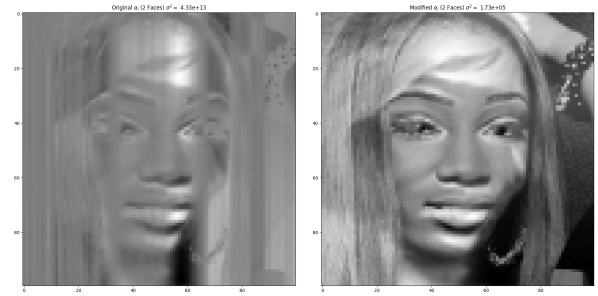

Fig. 35: Reconstructed faces: 2 eigenfaces (original-, modified weights)

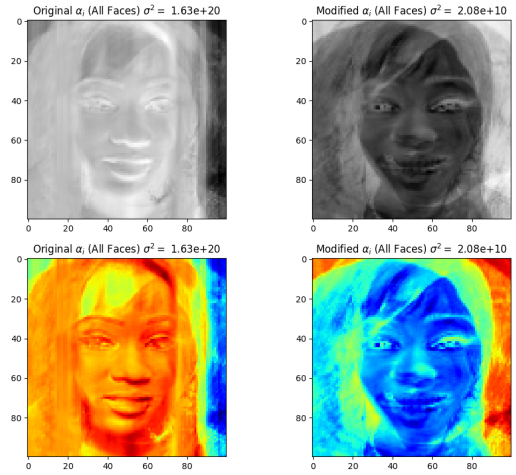

Fig. 36: Reconstructed faces: 10 eigenfaces (original-, modified weights)

Data set (OCL) – Person 4

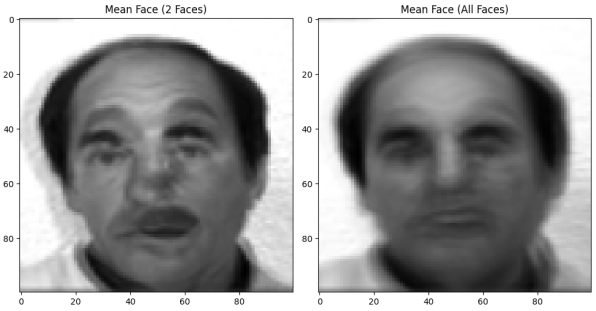

Fig. 37: Mean faces: 2 and $10(all)$ eigenfaces

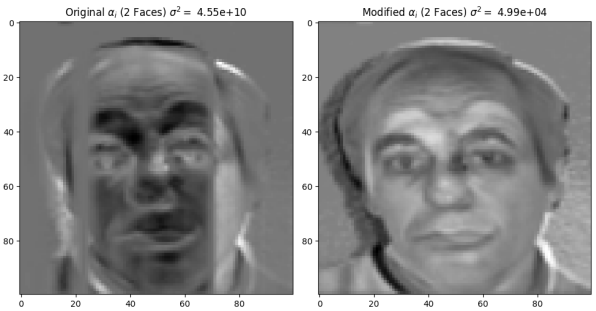

Fig. 38: Reconstructed faces: 2 eigenfaces (original-, modified weights)

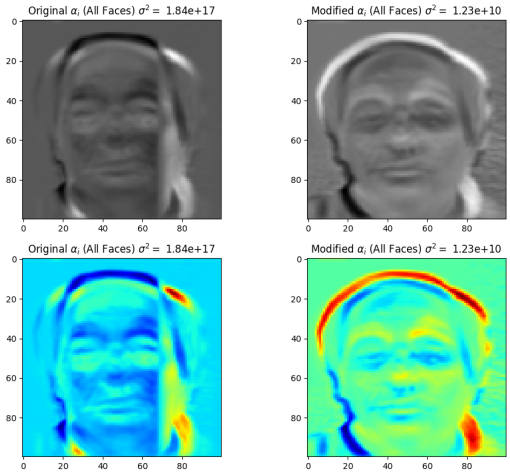

Fig. 39: Reconstructed faces: 10 eigenfaces (original-, modified weights)

Data set (OCL) – Person 10

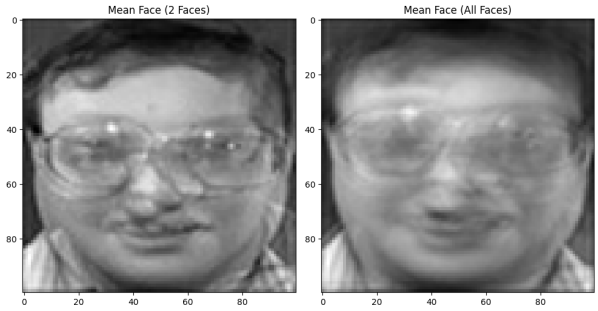

Fig. 40: Mean faces: 2 and $10(all)$ eigenfaces

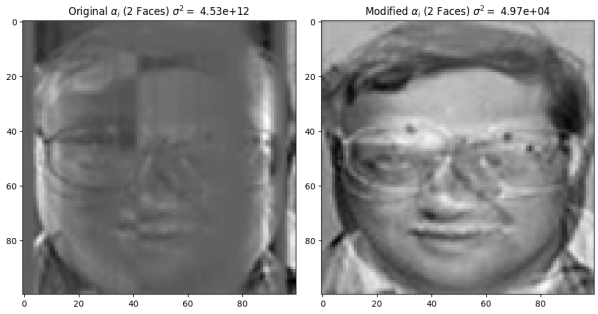

Fig. 41: Reconstructed faces: 2 eigenfaces (original-, modified weights)

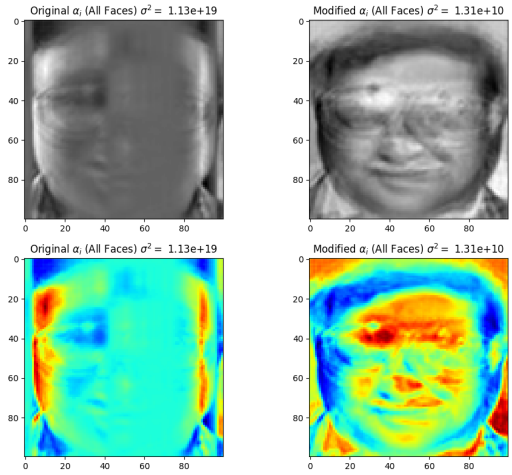

Fig. 42: Reconstructed faces: 10 eigenfaces (original-, modified weights)

Discussion Changing the weights, allows us to warp between the different eigenfaces/features. This fascinating process can cause changes in emotional expression or accentuation of facial characteristics for our subjects. For example, in figure 33, we can see a clear accentuating of the smile and cheekbones when using our modified weights.

In figure 39, we can observe, that the shape of the head and baldness is more accentuated using our modified weights. For our subject with glasses, , we were able accentuate the shape of his forehead, see figure 42.

When the faces are better aligned, the warping between eigenfaces looks smoother. We can conclude that the results can vary greatly between images, therefore it is recommend to test the reconstruction using a wide array of different images in regards to, lighting, facial expressions, alignment, etc.

5 Motion Energy Images (MEIs)

Background During the last task, we transfer from static images to videos. Videos are of course just images, presented at a sufficiently higher rate of change (frame rate). Our main concern will be investigating motion. We will look into optical flow, which is known as the apparent motion of objects in images between consecutive frames, which can be caused by movement of an object or the camera [15]. In this paper, we will use the Lucas-Kanade method, which is a differential method for optical flow estimation. The method is based on the assumption that the flow remains constant within the immediate neighborhood of the pixel of interest. It solves the fundamental optical flow equations for all the pixels within that neighborhood using the least squares criterion [16].

In order to start our optical flow computations, we first need to find the good points/features that are of real-world interest to track. To identify and track meaningful features, we used the Shi-Thomasi algorithm. The high-level idea intuition behind this algorithm, is that it computes the eigenvalues for each pixel of the image, and uses those to determine the corners with the highest corner response [17].

To expand our capabilities beyond capturing small motions, we investigated the use of image pyramids. Image

pyramids are used to improve the performance and robustness of tracking, particularly for larger translations [18]. This enables us to effectively handle larger motions. However, providing a detailed explanation of the workings of image pyramids is beyond the scope of this paper.

Implementation The following openCV methods were utilized [19] for our implementation:

- **cv.goodFeaturesToTrack()** - Shi-Tomasi algorithm for tracking good features.
- **cv.calcOpticalFlowPyrLK()** - Lucas-Kanade method with a pyramid approach.

Dataset Two videos were picked from the 'Actions as Space-Time shapes' dataset [20]. The first video is of a person 'one leg' jumping across the screen and the second video is of a person doing jumping jacks. Both videos contain a single person and a static camera. These videos are chosen because the action/motion is very explicit and relatively simple.

5.1 Optical Flow

As described above, we used various methods of *openCV* to capture the optical flow. The flow, was drawn based on a varying amount of frames w. By playing with the value of w, we could easily see which values were appropriate to capture our (partly) motion/action of interest!

The jumping jack seemed very interesting using $w = 5$ and $w = 15$ frame intervals. It also shows there is a slight difference in velocity, of the upward motion and downward motion in this action. The first figure nicely shows the 'full-motion' captured at 5 frame intervals. The second image, beautifully captures the downward motion at 15 frames. As said before, *Shi-Thomasi* was used to determine the 'good' features. For both images, the good features were correctly captured, since only those actual moving components, showed activity in the optical flow.

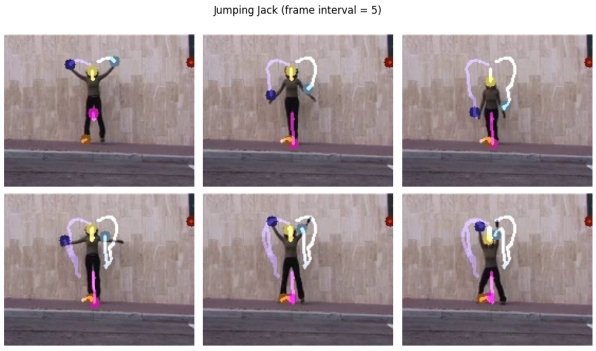

Fig. 43: Jumping jacks $w = 5$ - Notice how the first 5 images capture the full movement

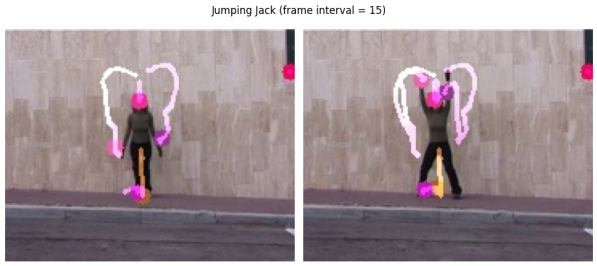

Fig. 44: Jumping jacks $w = 15$ – Downward motion

The second plot, shows how 10 frames allowed use to capture the 'action' of a person jumping across the screen. The same logical applies as to the previous capture.

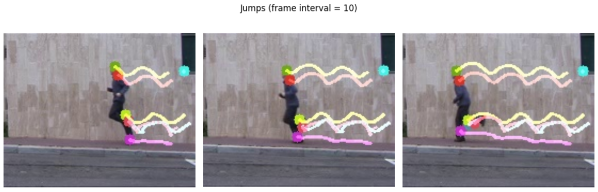

Fig. 45: Jumping $w = 10$ – 10 Frames capture the jump motion nicely

5.2 Motion Energy Images

'Impoverished motion data can evoke a strong precept'. A quote found on the internet by an unknown author. It does however, apply to the concept of Motion Energy Images (MEIs). We are interested in where motion is in the images, from which we can cumulate the motion to create motion energy image. Motion energy images are the **binarized** cumulative images of 'where stuff has moved' [21]. The frame interval w, plays a big role here in how closely each frame is connected to the previous. Which resembles how far we progressed in our motion. A high-level overview of the approach can be found below and is based on [22].

- Perform frame-by-frame subtraction *(image difference)*
- Accumulate the results over the history window w
- *Motion energy* is obtained by creating a binary version of the motion history.

Cleanup MEIs

After the MEIs were acquired, they were cleaned up using morphological operations. We decided to use dilation, since it is very effective with binary images [5] and in our case it allows use to fill the lattices.

Outline MEI

The outline of the MEI was found using *openCv* built-in **drawContours()** method. This allows for an easy and effective implementation with pleasing results.

Hu-Moments

The next step was to extract the shape descriptors of the MEIs. *openCv* built-in **HuMoments()** was used for this. In computer vision, image moments are used to characterize the shape of an object in an image [23]. Since we use the 'contours', we therefore, characterize our action/motion for subsequent frames.

Comparing the actions

As a final step, we can investigate how similar the 'jumping jacks' are to the odd-looking one-legged jumps, by making use of the previously mentioned Hu-moments. This activity recognition could be used in real-world surveillance applications, in order to distinguish between suspicious and normal behaviour [24].

This similarity of the hu-moments between different frames (intra action similarity) or different actions (inter action similarity) was measured by using mean squared error.

5.3 Results

Motion Energy Images

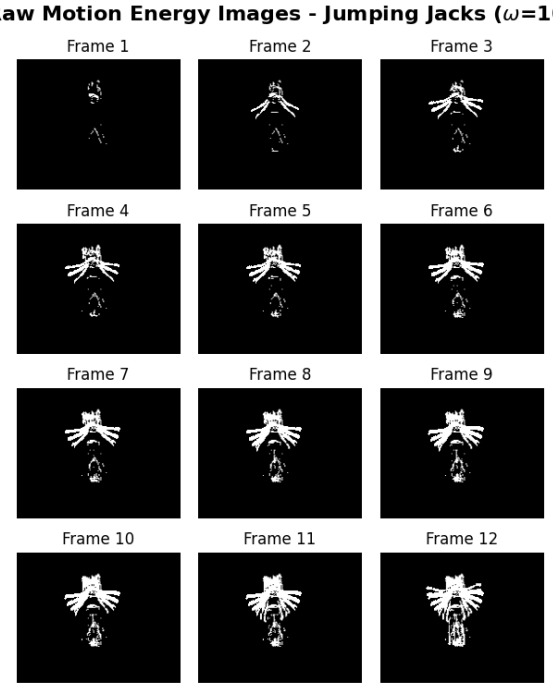

Fig. 46: Raw MEIs for Jumping Jacks

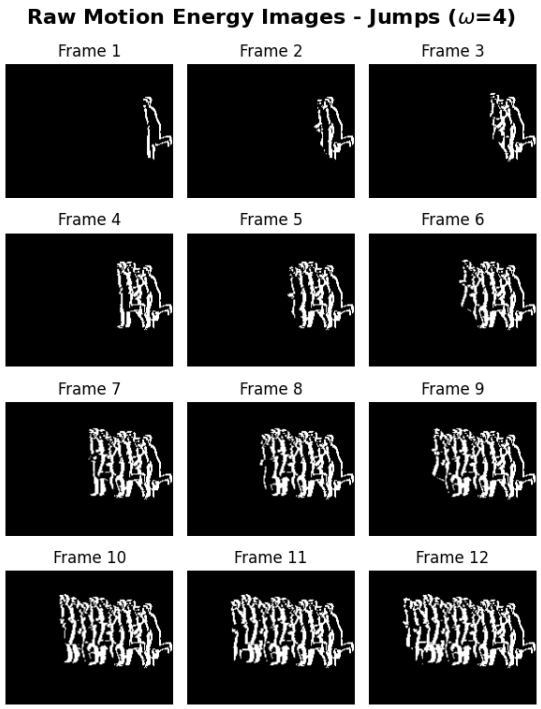

Fig. 47: Raw MEIs for Jumps

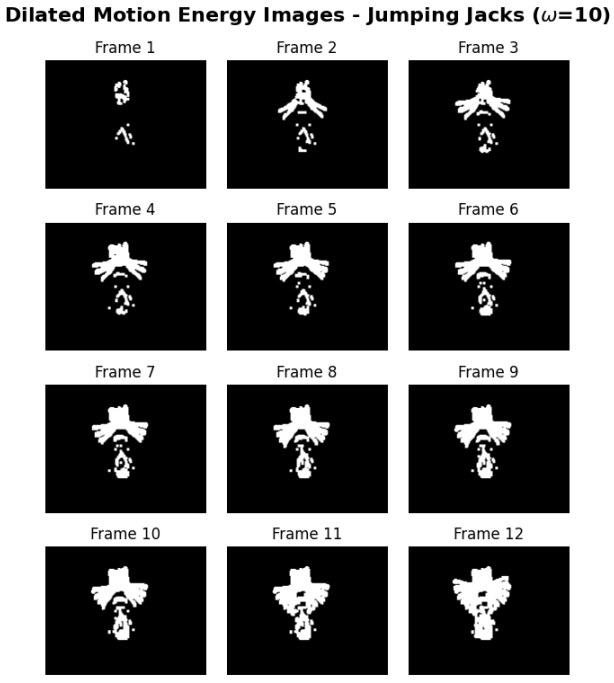

Fig. 48: Cleaned (dilated) MEIs for Jumping Jacks

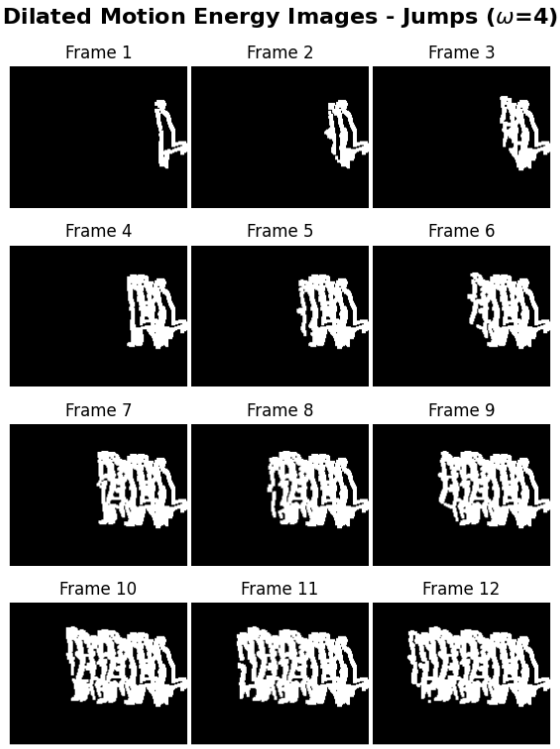

Fig. 49: Cleaned (dilated) MEIs for Jumps

Motion Outlines

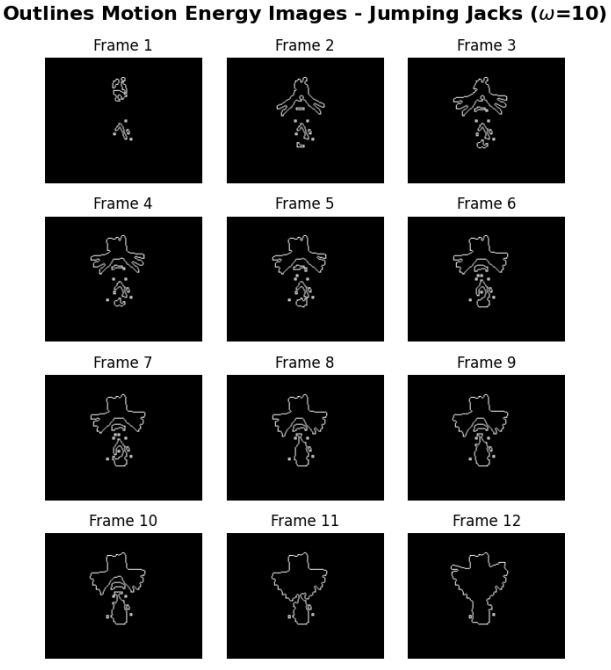

Fig. 50: Outline MEI Jumping Jacks

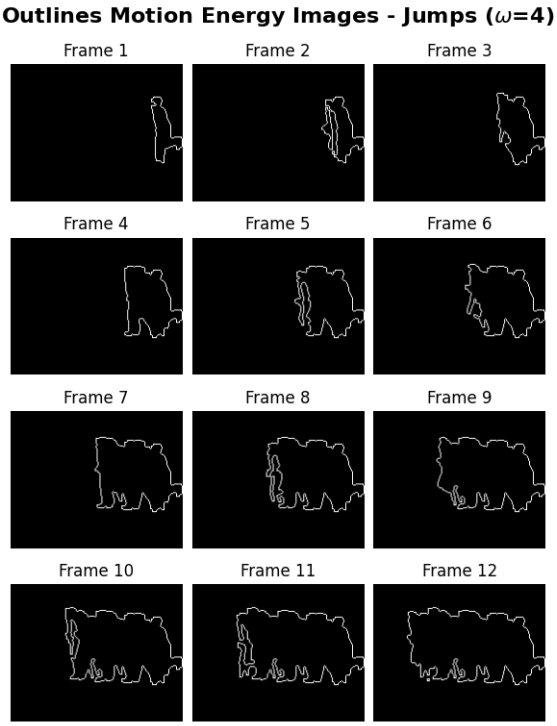

Fig. 51: Outline MEI Jumps

Intra Action Similarity

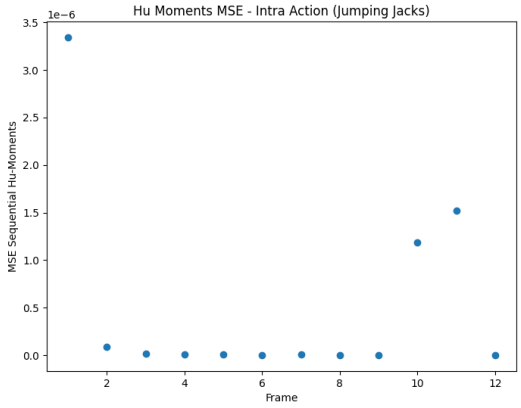

Fig. 52: MSE Hu-moments subsequent frames (Jumping Jacks)

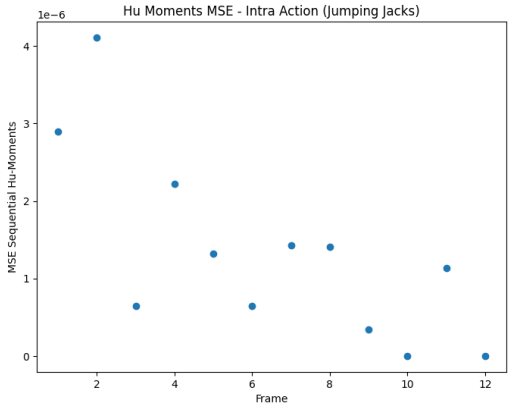

Fig. 53: MSE Hu-moments subsequent frames (Jumps)

Inter Action Similarity

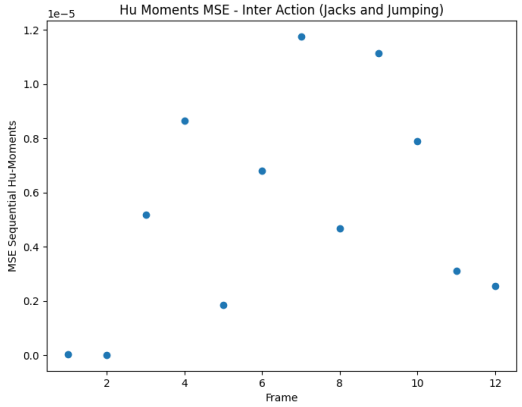

Fig. 54: MSE Hu-moments between two actions (Jumping Jacks and Jumps)

5.4 Discussion

The motion energy images, nicely showcase the binarized history energy images, therefore the cumulative motion. Since our values picked for w correspond relatively well to the motion, we could see nice outlines that characterise the action/motion of interest.

Figure 51, clearly shows a person jumping along the screen. Looking back at our quote mentioned before, it is fascinating how easily we can derive what 'action' is going from just a white outline on a black background.

As said before, Hu-moments can serve great value in determining the motion between frames and between actions. Comparing figure 51 and figure 53, seem to bring the same message. The motion of 'jupming' is relative similar between the frames, which can visually be observed in the motion energy images (MEIs) and reflects through the relative low values of MSE between hu-moments of sequential frames. In case of the jumping jacks, the MEIs look quiet different, which is in accordance with the graph.

As we have only compared two types of actions, it is challenging to draw any definite quantitative or qualitative conclusions regarding their similarity. To better understand the numerical values of the Mean Squared Error (MSE), it is important to investigate a broader range of actions. For instance, we could involve two individuals performing the same activity, such as "jumping jacks." Since each person may not move in exactly the same way but rather similarly, this would likely result in high similarity and consequently low MSE values. Exploring a wider variety of actions and their similarities would serve as an excellent initial step for further research.

Appendix

References

[1] Mei Wang, Weihong Deng, Jiani Hu, Xunqiang Tao, and Yaohai Huang. Racial faces in the wild: Reducing racial bias by information maximization adaptation network. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[2] AT&T Laboratories Cambridge. Database of faces. AT&T Laboratories Cambridge, 1994. Available at ftp://ftp.uk.research.att.com/pub/data/att_faces.tar.Z.

[3] Peeraya Sripian. Hybrid image as the assessment tool for myopia severity level. pages 85–85, 06 2017.

[4] Aude Oliva, Antonio Torralba, and Philippe Schyns. Hybrid images. *ACM Trans. Graph.*, 25:527–532, 07 2006.

[5] Rafael C. Gonzalez and Richard E. Woods. *Digital image processing*. Prentice Hall, Upper Saddle River, N.J., 2008.

[6] Steven W. Smith. *The Scientist and Engineer's Guide to Digital Signal Processing*. California Technical Publishing, 1997. Available at www.dspguide.com.

[7] M Tech and Ankur Saxena. Image watermarking using discrete cosine transform [dct] and genetic algorithm [ga] mahima singh. 06 2017.

[8] Keshav Rawat and Dheerendra Tomar. Digital watermarking schemes for authorization against copying or piracy of color images. *Indian Journal of Computer Science and Engineering*, 1, 12 2010.

[9] Larry L. Peterson and Bruce S. Davie. 7 - end-to-end data. In Larry L. Peterson and Bruce S. Davie, editors, *Computer Networks (Sixth Edition)*, The Morgan Kaufmann Series in Networking, pages 554–605. Morgan Kaufmann, sixth edition edition, 2022.

[10] S. Ri, H. Tsuda, K. Chang, S. Hsu, F. Lo, and T. Lee. Dynamic deformation measurement by the sampling moiré method from video recording and its application to bridge engineering. *Experimental Techniques*, 44, 01 2020.

[11] Cheng Tian, Ru-Hong Wen, Wei-Ping Zou, and Li-Hua Gong. Robust and blind watermarking algorithm based on dct and svd in the contourlet domain. *Multimedia Tools and Applications*, 79, 03 2020.

[12] M.A. Turk and A.P. Pentland. Face recognition using eigenfaces. In *Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 586–591, 1991.

[13] Andrzej Maćkiewicz and Waldemar Ratajczak. Principal components analysis (pca). *Computers Geosciences*, 19(3):303–342, 1993.

[14] LearnOpenCV. Face reconstruction using eigenfaces. https://learnopencv.com/face-reconstruction-using-eigenfaces-cpp-python/, Accessed: 2023.

[15] Deqing Sun, Stefan Roth, J. P. Lewis, and Michael J. Black. Learning optical flow. In David Forsyth, Philip Torr, and Andrew Zisserman, editors, *Computer Vision – ECCV 2008*, pages 83–97, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[16] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th international joint conference on Artificial intelligence - Volume 2*, pages 674–679. Morgan Kaufmann Publishers Inc., 1981.

[17] Jianbo Shi and Carlo Tomasi. Good features to track. *Proceedings / CVPR, IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 600, 03 2000.

[18] Jean-Yves Bouguet. Pyramidal implementation of the lucas kanade feature tracker description of the algorithm, 2000.

[19] Opencv optical flow tutorial. https://docs.opencv.org/3.4/d4/dee/tutorial_optical_flow.html. Accessed on 8 july 2023.

[20] Lena Gorelick, Moshe Blank, Eli Shechtman, Michal Irani, and Ronen Basri. Actions as space-time shapes. *Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2247–2253, December 2007.

[21] Aaron Bobick and J.W. Davis. The recognition of human movement using temporal templates. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23:257 – 267, 04 2001.

[22] Alexander Stoytchev. Motion energy & motion history. Class Lecture, February 2007. Slides for HCI/ComS 575X: Computational Perception, Iowa State University, Spring 2007.

[23] Zhihu Huang and J. Leng. Analysis of hu's moment invariants on image scaling and rotation. volume 7, pages V7–476, 05 2010.

[24] Wei Niu, Jiao Long, D. Han, and Yuanfang Wang. Human activity detection and recognition for video surveillance. pages 719 – 722 Vol.1, 07 2004.