Learning geometry using PCA and diffusion maps

Wemmenhove, A.J. (Jelle)

Award date:
2018

Learning Geometry using PCA and Diffusion Maps

Eindhoven University of Technology

Bachelor Final Project (2WH40)

Author
Jelle Wemmenhove

Supervisor
Dr. Jim Portegies

March 19, 2018

Abstract

Dimensionality reduction aims to uncover a small number of latent variables from high dimensional data. This report experimentally assesses the ability of PCA and diffusion maps to uncover the position and rotation of a robot equipped with a camera from only the images captured along its path. The results are promising, but there is still work to be done. Diffusion maps produce slightly better lower dimensional embeddings than PCA, but prior knowledge of the robot's movement is necessary to properly interpret them.

1 Introduction

High dimensional data is ubiquitous in the world. Whether you have a lot of sensors measuring the same phenomenon, such as radio telescopes, or are gathering different but related data, for example by doing a survey, you are working with high dimensional data. Unfortunately, it is hard to analyse such data, let alone visualise it. Part of its problems are caused by the counter-intuitive behaviour of higher dimensional spaces (Section 1.2.2 of [6] lists a large number of examples). One of the main goals of high dimensional data analysis is discovering a smaller number of variables hidden in the data that can fully explain our observations. To this aim, a multitude of dimensionality reduction techniques have been developed that attempt to remove the redundancy in the data.

Principal Component Analysis (PCA) is the oldest dimensionality reduction method, already created by Pearson in 1901 [11]. It assumes the data to be a linear transformation of some latent variables. In more recent years, a lot of non-linear dimensionality reduction methods were invented, including the method of diffusion maps by Coifman and Lafon [4], [1] in 2004. These methods perform much better on artificial datasets such as the 'Swiss roll' [6] than linear techniques, but this does not necessarily extend to real-world data [8]. To better compare dimensionality reduction methods, research is currently being done on assessing the quality of the lower dimensional embeddings produced, numerically [7], [10] and visually [9]. One approach is to apply dimensionality reduction techniques on data whose structure is known beforehand, such as trying to align images showing different head movements [6], [5].

In this report, we experimentally evaluate the performance of PCA and diffusion maps on real-world data. Using the programmable toy robot Cozmo developed by Anki, we collect 320×240 pixel images whilst having Cozmo perform several movements. We also look at the effects of pre-processing the images and of certain parameter choices. The images are fully defined by four latent variables constituting Cozmo's position and rotation. The aim of the experiments is to recover these variables from the images themselves.

Section 2 provides the mathematical background for understanding PCA and diffusion maps and specifies how they are applied to the images collected. In Section 3, we model the image collection process and apply the dimensionality reduction techniques to establish a framework against which to compare the results of the real-world images. Section 4 describes the methodology of the experiments, the results of which are presented and discussed in Section 5. We finish with a final evaluation of PCA's and diffusion maps' performance in Section 6.

2 Background

For the reader unfamiliar with the inner workings of PCA and diffusion maps, we present their full mathematical description. The analysis of the dimensionality reduction methods in Section 3 uses this description as well.

2.1 Principal Component Analysis

Intuitively, PCA fits a k-dimensional affine subspace to data points and considers the orthogonal projection of the data onto the subspace as the lower dimensional embedding. Due to the notion of fitting used by PCA, this is equivalent to first centring the data points by subtracting their average and then fitting a linear subspace to them.

PCA assumes the data to be elements of a m-dimensional Hilbert space H_m. Let $X \subset H_m$ be a multiset denoting the data. We limit ourselves to finite datasets, so $n := |X| \in \mathbb{N}$. Because the dataset is centred, we have that

$$\sum_{x \in X} x = 0 \ .$$

We aim to approximate X with an at most k-dimensional linear subspace $L_k \subset H_m$ such that the error

$$\sum_{x \in X} ||x - P_{L_k} x||^2$$

is minimised, with $P_L : H_m \to L$ the linear operator projecting vectors orthogonally onto a linear subspace $L \subset H_m$.

To determine L_k, we shift our perspective from spaces to linear operators and their images. Let $\mathbb{K}^X$ be the free vector space over X where $\mathbb{K}$ is a field. Let ι denote the canonical inclusion of X into $\mathbb{K}^X$. Consider the linear operator $Y : \mathbb{K}^X \to H_m$ such that $(Y \circ \iota)x = x$ and the linear operator $M_k : \mathbb{K}^X \to H_m$ that best approximates Y of all linear operators having rank at most k by minimising $||Y - M_k||_F$, where $||\cdot||_F$ denotes the Frobenius or Hilbert-Schmidt norm on linear operators. We claim that the linear subspace $M_k(\mathbb{K}^X)$ is L_k. Let $L'_k \subset H_m$ be any at most k-dimensional linear subspace. We have that L'_k is the range of the composition $P_{L'_k} Y : \mathbb{K}^X \to H_m$ which has rank at most k. This gives us that

$$\sum_{x \in X} ||x - P_{M_k(\mathbb{K}^X)} x||^2 \leq \sum_{x \in X} ||Y(\iota(x)) - M_k(\iota(x))||^2 = ||Y - M_k||_F \leq ||Y - P_{L'_k} Y||_F = \sum_{x \in X} ||x - P_{L'_k} x||^2 \ .$$

Thus, it suffices to find M_k. For this purpose, we consider the factorisation of Y using singular value decomposition (SVD) into

$$Y = U \Sigma V^*,$$

with $U : H_m \to H_m$ and $V : \mathbb{K}^X \to \mathbb{K}^X$ unitary operators and $\Sigma : \mathbb{K}^X \to H_m$ a scaling operator given by

$$\Sigma e_i = \begin{cases} \sigma_i f_i, & \text{if } i \leq \min\{n, m\} \\ 0, & \text{otherwise} \end{cases} \ ,$$

for orthonormal bases $\{e_1, \ldots, e_n\}$ of $\mathbb{K}^X$ and $\{f_1, \ldots, f_m\}$ of H_m where the so-called singular values σ_i of Y are non-negative real scalars such that

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{\min\{n,m\}} \geq 0 \ .$$

The scaled and rotated vectors $\sigma_j U f_j$ are the semi-axes of the ellipsodial image of the closed unit ball under Y. The Eckart-Young-Mirsky Theorem [2] states that M_k is equal to $U \Sigma_k V^*$ where $\Sigma_k : \mathbb{K}^X \to H_m$ is a rank k operator approximating Σ by setting all singular values following the k first ones to zero, i.e. M_k is a partial SVD of Y.

The semi-axes without scaling $Uf_1, \ldots, Uf_k$ form an orthonormal basis of L_k and are ordered in terms of importance: the first semi-axis is included in every approximating linear subspace $L_1, L_2, \ldots$, the second semi-axis is included in $L_2, L_3, \ldots$ and so on. Because of this, Uf_j is called the j-th principal component

of the data. The lower dimensional embedding of $x \in X$ is given by the orthogonal projection of x onto L_k:

$$P_{L_k} x = \sum_{j=1}^{k} \langle x, U f_j \rangle U f_j \ .$$

2.2 Diffusion maps

Diffusion maps assign distances between data points based on their 'affinity' and provide a way to approximate this geometry in a lower dimensional space using the Euclidean metric. The higher dimensional this space is taken to be, the better the approximation. However, to define the metric on the data takes some effort.

Let $(X, \mathcal{A}, \mu)$ be a measurable space with X a multiset denoting the data. We limit ourselves to finite datasets with $|X| = n$, hence there is no problem in putting $\mathcal{A} = 2^X$ and without loss of generality we assume that $\mu(x) := \mu(\{x\}) > 0$ for all $x \in X$. To create a diffusion map, one additional, but vital, ingredient is needed: the so-called kernel $k : X \times X \to [0, \infty)$ that represents the affinity between data points. The pair (X, k) induces a weighted graph on X with edges $\{(x, y) \in X \times X \mid k(x, y) > 0\}$ and weights given by k. We require that

(i) k is symmetric;

(ii) the graph defined by (X, k) is connected;

(iii) $k(x, x) > 0$ for all $x \in X$.

Each of these requirements can be interpreted as statements about the concept of affinity on the dataset. The main idea is that k sufficiently captures the geometry of the data, since this is the only way X's structure will be used.

From the graph defined by (X, k), we construct a Markov chain on X. This technique is known as the normalised graph Laplacian construction. Sections 6.1-6.6 of [3] provide an explanation of Markov chains and the techniques in this analysis. Let $d : X \to [0, \infty)$ be given by

$$d(x) = \int_X k(x, y) d\mu(y) \ .$$

The transition kernel $p : X \times X \to [0, 1]$ is then given by

$$p(x, y) = \frac{k(x, y)}{d(x)} \ ,$$

such that the probability $p(x, y)\mu(y)$ of jumping from a state $x \in X$ to another state $y \in X$ is proportional to the affinity between x and the elements in Y in relation to the total affinity felt by x.

The Markov chain constructed has three important properties that are to be explained. First of all, there exists a stationary distribution $\pi : X \to [0, 1]$ given by

$$\pi(x) = \frac{d(x)}{\sum_{x \in X} d(z)} \ ,$$

stating that the probability $\pi(x)$ of the random walk being in state $x \in X$ increases the more they are affiliated with the dataset. Because of requirement (ii) on k, the Markov chain is irreducible, meaning that there is a non-zero chance of reaching any vertex when starting from any other vertex. Any irreducible Markov chain has a unique stationary distribution that is invariant to steps taken by the random walk. A quick calculation shows that this is the case for π:

$$\int_X \pi(y) p(y, x) d\mu(y) = \int_X \frac{d(y)}{\sum_{x \in X} d(z)} \frac{k(y, x)}{d(y)} d\mu(y) = \frac{\int_X k(x, y) d\mu(y)}{\sum_{x \in X} d(z)} = \frac{d(x)}{\sum_{x \in X} d(z)} = \pi(x) \ .$$

Secondly, the Markov chain is reversible, meaning that the distribution π satisfies the detailed balance equations $\pi(x)p(x,y) = \pi(y)p(y,x)$ for all $x, y \in X$ because of the symmetry of k:

$$\pi(x)p(x,y) = \frac{d(x)}{\sum_{x\in X} d(z)} \frac{k(x,y)}{d(x)} = \frac{d(y)}{\sum_{x\in X} d(z)} \frac{k(y,x)}{d(y)} = \pi(y)p(y,x) \ .$$

Finally, the Markov chain is ergodic, i.e. it is positive recurrent and aperiodic. The Markov chain being positive recurrent follows from its irreducibility and the state space X being finite. The aperiodicity follows from requirement (i) on k.

Our analysis of the Markov chain revolves around the linear operator $P : L^2(X, d\mu) \to L^2(X, d\mu)$ given by

$$(Pf)(x) = \int_X p(x,y) f(y) d\mu(y) \ ,$$

the value of $f \in L^2(X, d\mu)$ you expect to encounter when taking a single step according to the random walk, starting at $x \in X$. We are specifically interested in he eigenvectors of P: functions whose value one expects to encounter equals the currently observed value up to multiplication by a scalar. Unfortunately, these are hard to find since P is not symmetric. We turn P into a symmetric linear operator A by conjugation with the bijective linear operator $\sqrt{\Pi} : L^2(X, d\mu) \to L^2(X, d\mu/\pi)$ given by

$$\left(\sqrt{\Pi} f\right)(x) = \sqrt{\pi(x)} f(x) \ .$$

This gives us the linear operator $A = \sqrt{\Pi} P \sqrt{\Pi}^{-1} : L^2(X, d\mu/\pi) \to L^2(X, d\mu/\pi)$ given by

$$(Af)(x) = \int_X a(x,y) f(y) d\mu(y)$$

where the kernel $a : X \times X \to \mathbb{R}$ is given by

$$a(x,y) = \sqrt{\pi(x)} p(x,y) \sqrt{\pi(y)}^{-1} = \frac{k(x,y)}{\sqrt{d(x)}\sqrt{d(y)}} \ .$$

The construction is illustrated in the commutative diagram

$$\begin{array}{ccc} L^2(X, d\mu) & \xrightarrow{P} & L^2(X, d\mu) \\ {\scriptstyle \sqrt{\Pi}^{-1}}\uparrow & & \downarrow{\scriptstyle \sqrt{\Pi}} \\ L^2(X, d\mu/\pi) & \xrightarrow{A} & L^2(X, d\mu/\pi) \end{array} \ .$$

A is indeed symmetric, therefore there exists an orthonormal basis of eigenvectors $\{\phi_1, \dots \phi_n\}$ of $L^2(X, \mu/\pi)$ with corresponding real eigenvalues $\{\lambda_1, \dots, \lambda_n\}$. We assume the eigenvalues to be sorted in decreasing order with respect to their absolute value:

$$|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_n| \ .$$

Composing $A\phi_\ell = \lambda_\ell \phi_\ell$ on the left-hand side with $\sqrt{\Pi}^{-1}$, shows that $\psi_\ell := \sqrt{\Pi}^{-1} \phi_\ell$ for $\ell = 1, \dots, n$ are eigenvectors of P with the same eigenvalues:

$$P\sqrt{\Pi}^{-1} \phi_\ell = \sqrt{\Pi}^{-1} A \phi_\ell = \sqrt{\Pi}^{-1} \lambda_\ell \phi_\ell = \lambda_\ell \sqrt{\Pi}^{-1} \phi_\ell \ .$$

We now investigate the eigenvalues. Analysing A provides us with an upper limit. Consider $\langle Af, f \rangle$ for any $f \in L^2(X, d\mu/\pi)$:

$$\langle Af, f \rangle = \int_X \int_X k(x,y) \frac{f(x)}{\sqrt{d(x)}} \frac{f(y)}{\sqrt{d(y)}} d\mu(x) d\mu(y) \ .$$

Using Cauchy-Schwarz' inequality, we find that

$$\begin{aligned} \left| \int_X k(x,y) \frac{f(y)}{\sqrt{d(y)}} d\mu(y) \right| &\leq \left(\int_X k(x,y) d\mu(y) \right)^{1/2} \left(\int_X k(x,y) \frac{f(y)^2}{d(y)} d\mu(y) \right)^{1/2} \\ &= \sqrt{d(x)} \left(\int_X k(x,y) \frac{f(y)^2}{d(y)} d\mu(y) \right)^{1/2} \end{aligned}$$

such that

$$|\langle Af, f\rangle| \leq \int_X |f(x)| \left(\int_X k(x,y) \frac{f(y)^2}{d(y)} d\mu(y) \right)^{1/2} d\mu(x) \ .$$

Applying the Cauchy-Schwarz' inequality a second time, we find that due to the symmetry of k

$$\langle Af, f\rangle \leq ||f|| \left(\int_X \int_X k(x,y) \frac{f(y)^2}{d(y)} d\mu(y) d\mu(x) \right)^{1/2} = ||f||^2 \ ,$$

implying that $|\lambda_1| \leq 1$. Moreover, $\lambda_1 = 1$ since $P\mathbf{1} = \mathbf{1}$ where $\mathbf{1} : X \to \{1\}$. But what about the other eigenvalues? Consider a basis of $L(X, d\mu)$ consisting of indicator functions $\{1_x\}_{x \in X}$. This allows us to represent P^t with $t \in \mathbb{N}$ as a matrix where the elements are of the form $\langle P^t 1_y, 1_x\rangle$ which can be rewritten as

$$\mu(x) \sum_{s_1 \in X} p(x, s_1)\mu(s_1) \left(\sum_{s_2 \in X} p(s_1, s_2)\mu(s_2) \left(\dots \left(\sum_{s_{t-1} \in X} p(s_{t-2}, s_{t-1})\mu(s_{t-1}) p(s_{t-1}, y)\mu(y) \right) \dots \right) \right) \ ,$$

that is the probability of reaching $y \in X$ from $x \in X$ in t steps starting in x, multiplied with $\mu(x)$. Since the Markov chain is ergodic, there exists a power of P such that its matrix representation with respect to the indicator basis contains only positive elements. The Perron-Frobenius theorem states that because of this, λ_1 has a multiplicity of one and that all other eigenvalues are strictly smaller than λ_1 in absolute value. Thus, we have that

$$1 = \lambda_1 > |\lambda_2| \geq \dots \geq |\lambda_n| \ .$$

We are now able to define the metric on the dataset X! Let the diffusion metric $D : X \times X \to [0, \infty)$ be given by

$$D(x,y)^2 := ||p(x,\cdot) - p(y,\cdot)||^2_{L^2(X, d\mu/\pi^2)} = \int_X (p(x,z) - p(y,z))^2 \frac{d\mu(z)}{\pi(z)^2} \ .$$

The transition kernel p can be characterised in terms of the eigenvalues and eigenvectors. Since the eigenvectors $\phi_1, \dots, \phi_n$ form an orthonormal basis of $L^2(X, d\mu/\pi)$, we have that

$$p(x,y) = \sqrt{\pi(x)}^{-1} a(x,y) \sqrt{\pi(y)} = \sqrt{\pi(x)}^{-1} \left(\sum_{\ell=1}^n \lambda_\ell \phi_\ell(x) \phi_\ell(y) \right) \sqrt{\pi(y)} = \sum_{\ell=1}^n \lambda_\ell \psi_\ell(x) \varphi_\ell(y)$$

where $\varphi_\ell = \sqrt{\Pi}\phi_\ell$ for $\ell = 1, \dots, n$. This results in

$$D(x,y)^2 = \left|\left| \sum_{\ell=1}^n \lambda_\ell (\psi_\ell(x) - \psi_\ell(y)) \varphi_\ell \right|\right|^2_{L^2(X, d\mu/\pi^2)} = \sum_{\ell=2}^n \lambda_\ell^2 (\psi_\ell(x) - \psi_\ell(y))^2 \ ,$$

since $\{\varphi_1, \dots, \varphi_n\}$ forms an orthonormal basis of $L^2(X, d\mu/\pi^2)$. Note that we left the first eigenvalue λ_1 and eigenvector ψ_1 out of the summation, since ψ_1 is the constant function $\mathbf{1}$. Thus, if we embed the data via the diffusion map $\Psi_m : X \to \mathbb{R}^m$ given by

$$\Psi_m(x) = (\lambda_2 \psi_2(x), \dots, \lambda_{m+1} \psi_{m+1}(x)) \ ,$$

we have that $||\Psi_m x - \Psi_m y||$ approximates $D(x,y)$ better and better for increasing m. The larger eigenvalues are used first to close the gap as quickly as possible.

This is all nice and dandy, but how should one obtain the kernel k? Since we want k to represent the geometry of the data, it seems only logical to take $k(x,y)$ to be the value of a positive, decreasing function $h : [0, \infty) \to (0, \infty)$ on the distance $d(x,y)$ where d can be any metric on X. We suggest scaling $d(x,y)$ by a constant $\rho > 0$ such that the mean distance on X multiplied by ρ can be used as a parameter: the rescaled average distance $R \geq 0$. This allows the scope of the local geometry to be altered and it produces similar kernels when the distances change due to choice of metric or dimensionality of the dataset. We use the function h given by $h(x) = \exp(-x)$ resulting in the rotationally invariant kernel k_R given by

$$k_R(x,y) = \mathrm{e}^{-\rho d(x,y)} \ .$$

We are interested in uncovering the underlying geometry manifestating as the data points, but their distribution over the geometry does not have to be uniform. Certain parts of the geometry might be better represented than others. To decouple geometry and density, [1] suggests using an anisotropic kernel by dividing the rotationally invariant kernel k_R by notions of the total affinity of the two data points. Let $\alpha \in [0,1]$ be a parameter controlling the anisotropicity of the kernel $k_R^{(\alpha)}$ given by

$$k_R^{(\alpha)}(x,y) = \frac{k_R(x,y)}{\left(\int_X k_R(x,z)d\mu(z)\right)^\alpha \left(\int_X k_R(y,z)d\mu(z)\right)^\alpha} \ .$$

2.3 Images

Although the dimensionality reduction methods are expressed in terms of general datasets, we are interested in applying them to images. We model images as mappings from a canvas C to a range of values V. Cozmo's images are discrete, they map a discrete grid of pixels $\{1,\dots,N\} \times \{1,\dots,M\}$ with N and M are the number of pixels in horizontal and vertical direction respectively, to G discrete, evenly spaced shades of grey $\{0/(G-1), 1/(G-1), \dots (G-1)/(G-1)\}$ with 0 and 1 representing white and black respectively. This set of mappings is denoted by $\{0, 1/(G-1), \dots, 1\}^{\{1,\dots,N\}\times\{1,\dots,M\}}$ and is a subset of the free vector space $\mathbb{R}^{\{1,\dots,N\}\times\{1,\dots,M\}}$ which constitutes a MN-dimensional Hilbert space.

PCA requires that the data be centred by subtracting their center of mass, but this can result in centred mappings no longer being interpretable as images because they might map to values outside the permitted range. So in order to represent any vector $\varphi \in \mathbb{R}^{\{1,\dots,N\}\times\{1,\dots,M\}}$ as an image, we have to transform them. Let $\min\varphi$ and $\max\varphi$ denote the minimum and maximum value of φ over $\{1,\dots,N\} \times \{1,\dots,M\}$. If $\min\varphi \neq \max\varphi$, we first apply the mapping $T : \mathbb{R}^{\{1,\dots,N\}\times\{1,\dots,M\}} \to [0,1]^{\{1,\dots,N\}\times\{1,\dots,M\}}$ given by

$$(T\varphi)x = \left(\frac{\varphi(x) - \min\varphi}{\max\varphi - \min\varphi}\right) \ .$$

Consecutively, φ is transformed by rounding every value $\varphi(x)$ for $x \in \{1,\dots,N\} \times \{1,\dots,M\}$ to the closest value in $\{0, 1/(G-1), \dots, 1\}$. Note that when vectors are transformed by applying T, their range of values is spread out over $[0,1]$ such that their image always contains one white and one black pixel. For PCA, the transformation allows us to represent the projection of an image onto the affine subspace as as an image in itself, showing how well the principal components are able to encode the original image. It is also possible to represent the principal components themselves as images, either directly or translated by adding the center of mass of the original images to them.

For diffusion maps, we just use the counting measure μ_c. Sadly, it is not possible to represent the two-dimensional projections as images as we did with PCA, since a diffusion maps is an explicit mapping instead of an implict one such as PCA [6].

3 Theoretical exploration

Before performing experiments using Cozmo, we investigate the possible results we could get. This provides a framework against which to compare the actual results to see how well the dimensionality reduction performs in practice.

3.1 Mathematical model

We start by providing a mathematical description of the camera images captured by Cozmo, our objects of interest.

3.1.1 The model

The experiments performed are limited to Cozmo sitting on top of an empty table in an unchanging environment. The images captured are then fully determined by its position and viewing direction. As Cozmo is equipped with edge-detection, its position is limited to the surface of the table top, which can be represented by the subset $T \subset \mathbb{R}^2$. Cozmo's viewing direction can also be described by two parameters: the amount Cozmo's body is spun around a vertical axis pointing up from Cozmo's position in reference to a fixed point and the amount Cozmo's head is tilted in reference to the direction of the table top's surface. The angle of Cozmo's viewing direction in reference to a fixed point is naturally represented by the unit circle $\mathbb{S}^1$, since it is unrestrained and periodical. By design, Cozmo's head tilt is restricted in degrees to the interval $I := [-25.00, 44.50]$. Thus, Cozmo's freedom of movement can be described by the four-dimensional manifold

$$M := T \times \mathbb{S}^1 \times I \ .$$

Next, we need to describe how an element in the manifold M is transformed into a camera image. This process consists of two distinct steps expressed as mappings: a projection P from the three-dimensional world onto a canvas and a discretisation D from a continuous image to a pixelated image. Images are modelled as described in Section 2.3 that also details the nature of the discrete images. The continuous images in the codomain of the projection P are mappings from a rectangular canvas $[0, W] \times [0, H]$ with W and H the width and height of the images, to the interval $[0, 1]$ that represents shades of grey with 0 and 1 denoting white and black respectively. The transformation to camera images can now be written as

$$M = T \times \mathbb{S}^1 \times I \quad \xrightarrow{P} \quad [0,1]^{[0,W]\times[0,H]} \quad \xrightarrow{D} \quad \{0, 1/(G-1), \ldots, 1\}^{\{1,\ldots,N\}\times\{1,\ldots,M\}} \ ,$$

and our objects of interest are the images of M under the composition $D \circ P$. We take a very simple, idealised model for Cozmo's camera. Otherwise, there are too many subtleties to consider, and moreover, the map P is much more complicated than any reasonable choice of D, as will be shown. P's inner workings are as follows: Cozmo's position specifies a viewpoint which acts as the eye of an observer looking at the world and which part of the world is visible to the observer depends on the viewing direction specified by Cozmo's rotation. The canvas $[0, W] \times [0, H]$ is the intersection between a plane orthogonal to the viewing direction and the half-lines originating at the viewpoint within the field of view. As light reaches the viewpoint via one of these half-lines, each half-line can be assigned a brightness level. To produce the image, each point on the canvas is mapped to a number in the interval $[0, 1]$ by representing the brightness level of the half-line intersecting the point as a shade of grey. The projection is illustrated in Figure 1.

This model is already too complicated for testing the dimensionality reduction techniques and a mathematical analysis also leads to a dead end. For example, we would like P to be continuous, and if the reader were to look at the world and move their head around ever so slightly, it would seem that it is. Making this mathematically sound however is a difficult task. To be able to talk about continuity in the first place, we need to endow the function space $[0, 1]^{[0,W]\times[0,H]} =: F$ with a topology, for which there is no obvious choice. But for argument's sake, let us consider F as a L^2-space equipped with the Lebesgue measure and assume that the images are measurable functions. With this added structure, it could be argued that the projection is continuous when the view point is kept fixed, but being allowed to move the view point allows us to create a counter example.

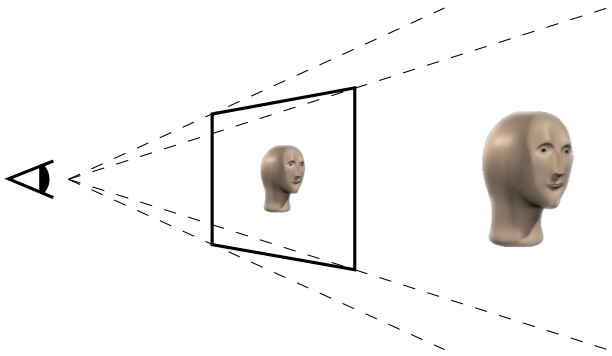

Figure 1: Illustration of the projection P that projects the real-world onto a flat canvas.

Counterexample 3.1. *For the environment, consider a closed box missing a single point. Furthermore, let the brightness level of the half-line be converted to black on the canvas if the half-line intersects with the box and let the brightness level be converted to white otherwise. (The box acts as a black body illuminated from all sides with a uniform light intensity.) If we take the view point to be the missing point of the box and have the viewing direction point away from the box, the image produced will be completely white. However, if the view point is moved inside the box by any non-zero amount, the image produced is black except for one white dot, the intersection of the canvas with the half-line going through the missing point. As a result, the images are separated by a distance of WH, despite the movement being arbitrarily small.*

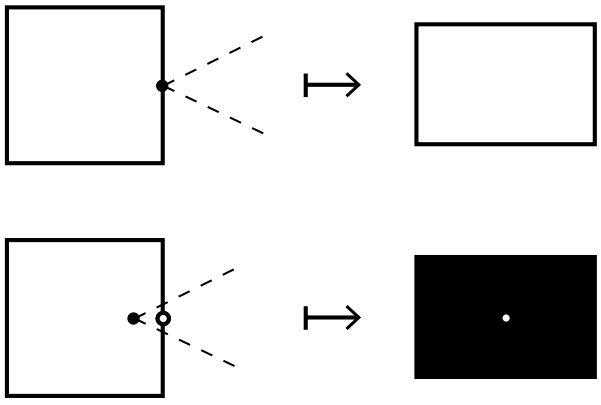

Figure 2: Illustration of the counterexample disproving continuity.

This counterexample is unrealistic as it heavily relies on the mathematical concept of points and the existence of point-sized camera's, but here also lies the difficulty in arguing that the projection is continuous, that is to come up with a more realistic model that is still simple enough to proof such properties in their full generality.

Despite the model providing a mathematical formulation of the images captured by Cozmo, it has failed to generate insight in the results produced by the experiments. This leads us to explore a simpler model.

3.2 Simplified model

A simpler model is achieved by skipping over the collection process and directly modelling the images produced. This distances our model from reality, but hopefully the model remains a good representation. We restrict the model to the images produced by rotating the camera horizontally. Because of this, it

suffices to model the images as being one pixel high. For simplicity, let us only consider images captured such that the pixels in the previous image are shifted one position to the right in the next image. This way, new pixelvalues are introduced only in an image's left-most pixel during the first full rotation being completed. The unchanging environment can thus be represented by the pixelvalues as detected during a full rotation. With each rotation, the same pixelvalues are encountered, hence this periodic nature is to be included in the model.

All-in-all, we arrive at the following simplified model. Let Cozmo be rotated at fixed speed and for every unit of time that passes, Cozmo takes an image. These images are such that the pixels in the previous image are shifted one position to the right in the next image. The environment is then represented by a function $e : \mathbb{Z}/M\mathbb{Z} \to \{0, 1/(G-1), \ldots, 1\}$, with $M \in \mathbb{N}$ the number of images captured during a full rotation and G the number of available pixelvalues. The image taken after $k \in \mathbb{N}$ units of time is modelled by a function $\mathrm{im}_k : \{1, \ldots, N\} \to \{0, 1/(G-1), \ldots, 1\}$, with $N \in \mathbb{N}$ the number of pixels in an image ($N \leq M$), given by

$$\mathrm{im}_k(\ell) = e((\ell - k) \mod M) \ .$$

Now this is a model we can work with.

3.3 Expected results

We use the simplified model from Section 3.2 to see how well the dimensionality reduction techniques Section 2 perform on several test environments. We reduce the images to two-dimensional data points, plotting their first coordinate along the horizontal axis and their second coordinate along the vertical axis when visualising the embeddings.

3.3.1 Initial test

For the first test, consider the test environment $e_1 : \mathbb{Z}/M\mathbb{Z} \to \{0, 1/(G-1), \ldots, 1\}$ given by

$$e_1([i]) = \begin{cases} 1, & \text{if } i = 0 \\ 0, & \text{otherwise} \end{cases}$$

and the images $\mathrm{im}_1, \ldots, \mathrm{im}_M : \{1, \ldots, N\} \to \{0, 1/(G-1), \ldots, 1\}$ generated by it. e_1 represents a bright environment with a single black marking, see Figure 3.

Figure 3: Illustration of the test environment e_1, note that the periodical nature of e_1 is lost.

3.3.2 PCA's performance on the initial test

PCA is applied to the centred images, i.e. $\{\mathrm{im}_k - (1/M)\mathbf{1}\}_{k=1}^{M} =: X$. Unfortunately, it fails to properly preserve their structure when reducing their dimensionality. First, let us consider the case when $M = N$. The images after centring $(\mathrm{im}_1 - (1/N)\mathbf{1}), \ldots, (\mathrm{im}_N - (1/N)\mathbf{1})$ span a $(N-1)$-dimensional linear subspace L orthogonal to the constant function $\mathbf{1} : \{1, \ldots, N\} \to \{1\}$. Because of symmetry, this results in the closed unit ball in $\mathbb{R}^X$ being mapped to a closed unit ball in $\mathbb{R}^{\{1,\ldots,N\}}$ flattened in the direction of 1. Thus, when PCA chooses two dimensions to represent the data, it ignores $N-3$ equally important ones.

But what if $M > N$? What is the effect of including images without a marking? Most importantly, the first N images are no longer pulled into the linear subspace L by centring since its effect has been weakened. This results in ellipsoidal image of the closed unit ball in $\mathbb{R}^X$ no longer being flattened in the direction of $\mathbf{1}$. The subset of the ellipsoid orthogonal to $\mathbf{1}$ remains unchanged however, as exemplified by the vector

$$v_1 = \frac{\sqrt{N-1}}{N\sqrt{N}} 1_{\mathrm{im}_1} - \frac{1}{N\sqrt{N}\sqrt{N-1}} 1_{\mathrm{im}_2} - \ldots - \frac{1}{N\sqrt{N}\sqrt{N-1}} 1_{\mathrm{im}_N} \in \mathbb{R}^X$$

with $1_{\mathrm{im}_k} : X \to \mathbb{R}$ an indicator function mapping $(\mathrm{im}_k - (1/M)\mathbf{1})$ to 1 and other elements of X to zero. v_1 lies on the edge of the closed unit ball in $\mathbb{R}^X$ and gets mapped to a normalisation of the vector $(\mathrm{im}_1 - (1/N)\mathbf{1})$ that helped span the linear subspace L from before. Similarly, we can construct vectors $v_2, \ldots, v_N \in \mathbb{R}^X$ that get mapped to $(\mathrm{im}_2 - (1/N)\mathbf{1}), \ldots, (\mathrm{im}_N - (1/N)\mathbf{1})$ respectively. Now all we need to fully determine the ellipsoid is the length a of its semi-axis in the direction of $\mathbf{1}$. The vector $(\mathrm{im}_1 - (1/M)\mathbf{1})$ lies on its boundary and using the normalisation of $(\mathrm{im}_1 - (1/N)\mathbf{1})$ as one of the semi-axis of the ellipsoid, $(\mathrm{im}_1 - (1/M)\mathbf{1})$ can be written as a linear combination of the normalised semi-axis

$$\mathrm{im}_1 - (1/M)\mathbf{1} = \frac{\sqrt{N-1}}{\sqrt{N}} \frac{\mathrm{im}_1 - (1/N)\mathbf{1}}{||\mathrm{im}_1 - (1/N)\mathbf{1}||} + \frac{M-N}{M\sqrt{N}} \frac{\mathbf{1}}{||\mathbf{1}||} \ .$$

Using the scalars in the linear combination above in the standard coordinate formula for ellipsoids, we find that

$$\frac{N-1}{N} + \frac{\left(\frac{M-N}{M}\right)^2 / N}{a^2} = 1 \ ,$$

resulting in $a = (M-N)/M$. Thus, the inclusion of unmarked images increases the dimensionality of the centred dataset, but the added dimension always is the least important one. The added structure does not solve PCA's problem of having to choose between $N-1$ equally important dimensions.

To gain some insight into the way PCA performs on the test environment e_1, we look at the dimensionality reduction of the first M images produced by putting $M = 7$ and $N = 5$. Figure 4 shows the images projected onto a two-dimensional space by PCA. Image im_1 is assigned its own dimension, whilst the other images with a marking have to share the remaining dimension.

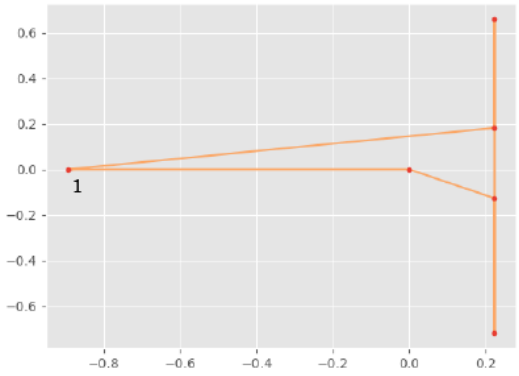

Figure 4: Two-dimesional embedding of the images produced by the test environment e_1 using PCA.

As described in Section 2.3, we can represent the two principal components identified by PCA as images. We chose to first undo the subtraction of the center of mass $(1/7)\mathbf{1}$. The results are shown in Figure 5.

Figure 5: Representation of the principal components identified by PCA for test environment e_1 as images.

Figure 5 shows that the first principal component is the opposite of the first image im_1. The second component seems to be a mixture of the third and fourth images im_3 and im_4, but it also captures the other images $\mathrm{im}_1, \mathrm{im}_2$ and im_5 in varying degrees. To see how well these principal components are able to represent the images, Figure 6 shows the orthogonal projections of the images onto the linear subspace spanned by the principal components as images, again having first undone the centring.

As expected, the dimensionality of image im_1 is reduced well. Unfortunately, the lower dimensional embedding is not so kind to the images $\mathrm{im}_2, \mathrm{im}_3, \mathrm{im}_4$ and im_5. The black marking is smeared out over

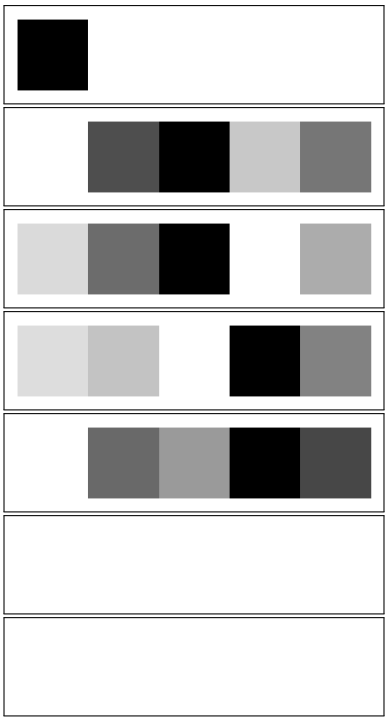

Figure 6: Representation of the projections of $\mathrm{im}_1, \ldots, \mathrm{im}_7$ (from top to bottom) as images.

multiple pixels and the projections of im_2 and im_5 seem to fit the images im_3 and im_4 better respectively. The projections of the images im_6 and im_7 without the marking resemble the original images quite well, since the zero-vector is very close to the function $(1/7)\mathbf{1}$ used in the representation.

3.3.3 Diffusion maps' performance on the initial test

For the initial test environment e_1, diffusion maps suffer from the same problems as PCA, it has to select two dimensions from $N-1$ equally important ones. Diffusion maps are equipped with a lot of parameters to tweak the final outcome, but this complicates the analysis partially. For simplicity, we stick with using the 1-norm to compute distances between images and we put $\alpha = 0$. We leave the rescaled average distance $R \geq 0$ undefined, using the corresponding scaling factor $\rho > 0$ in our formula's. The method of diffusion maps is applied to the set of images $X := \{\mathrm{im}_1, \ldots, \mathrm{im}_n\}$.

Again, we first consider the case where $M = N$, where all vertices behave similarly as shown by the transition kernel $p : X \times X \to [0, 1]$ that also represents transition probabilities is given by

$$p(x, y) = \begin{cases} A, & \text{if } y = x \\ A\mathrm{e}^{-2\rho}, & \text{otherwise} \end{cases},$$

where $A = (1+(N-1)\mathrm{e}^{-2\rho})^{-1}$ is a normalisation factor. These probabilities define the transition operator $P : L^2(X, d\mu_c) \to L^2(X, d\mu_c)$ for which we attempt to find a linearly independent set of eigenvectors spanning $L^2(X, d\mu_c)$. As always, $\mathbf{1} : X \to \{1\}$ is the first eigenvector. Next, we try some simpler function. No function that maps only a single element from X to a non-zero value can be an eigenvector of P, so we consider the a slightly more complex function. Let $f := f_{(x,y)} : X \to \mathbb{R}$ be a function mapping two distinct vertices $x, y \in X$ to non-zero values and every other vertex $z \in X \setminus \{x, y\}$ to zero. We want that $Pf = \lambda f$ for some $\lambda \in \mathbb{R}$ which comes down to the following system of equations

$$\begin{cases} p(x,x)f(x) + p(x,y)f(y) &= \lambda f(x) \\ p(y,y)f(y) + p(y,x)f(x) &= \lambda f(y) \\ p(z,x)f(x) + p(z,y)f(y) &= 0 \end{cases}, \tag{1}$$

where $z \in X \setminus \{x, y\}$. The final equation implies that $f(x) = -f(y)$, which unifies the first two equations

into

$$Af(x) - A\mathrm{e}^{-2\rho} f(x) = \lambda f(x) \ ,$$

which is satisfied by every value for $f(x)$ when $\lambda = A(1 - \mathrm{e}^{-2\rho})$. For simplicity, we take $f(x) = 1$, such that $f = f_{(x,y)}$ is an eigenvector of P with eigenvalue $A(1 - \mathrm{e}^{-2\rho})$. Similarly we can construct $N - 2$ other eigenvectors $f_{(x,z)}$ with $x \in X$ fixed and $z \in X \setminus \{x, y\}$ forming a linearly independent set. Note that each of these eigenvectors are orthogonal to $\mathbf{1}$ such that

$$L^2(x, d\mu_c) = \mathrm{span}\{\mathbf{1}\} \oplus \mathrm{span}\{f_{(x,z)}\}_{z \in X \setminus \{x\}} \ .$$

Thus there exists a orthonormal basis of $L^2(X, d\mu_c)$ of eigenvectors of P where the first eigenvalue is 1 and all the other eigenvalues are $A(1 - \mathrm{e}^{-2\rho})$. Since the first eigenvector is never used, two dimensions need to be chosen from $N - 1$ equally important ones to represent the diffusion distances on X in an Euclidean space.

Next, we look at the effect of including the unmarked images. For notation's sake, we split the images into two disjoint sets: $V = \{\mathrm{im}_1, \ldots, \mathrm{im}_N\}$ and $W = \{\mathrm{im}_{N+1}, \ldots, \mathrm{im}_M\}$. The transition kernel $p : X \times X \to [0, 1]$ is then given by

$$p(x, y) = \begin{cases} B, & \text{if } y = x \\ B\mathrm{e}^{-2\rho}, & \text{if } y \in V \setminus \{x\} \\ B\mathrm{e}^{-\rho}, & \text{if } y \in W \end{cases} \ ,$$

for $x \in V$ with the normalisation factor $B = (1 + (N - 1)\mathrm{e}^{-2\rho} + (M - N)\mathrm{e}^{-\rho})^{-1}$ and by

$$p(u, v) = \begin{cases} C, & \text{if } v \in W \\ C\mathrm{e}^{-\rho}, & \text{if } v \in V \end{cases} \ ,$$

for $u \in W$ with $C = ((M - N) + N\mathrm{e}^{-\rho})^{-1}$. Note that when starting in a vertex $u \in W$, the probability of jumping to any $v \in W \setminus \{u\}$ equals the probability of jumping to itself, as opposed to the transition probabilities for images in V. As always, the constant function $\mathbf{1}$ is an eigenvector to the transition operator P defined by the transition kernel. More surprisingly, the eigenvectors $f_{(x,y)}$ from before are eigenvectors as well (when taking $f_{(x,y)}(u) = 0$ for all $u \in V$), albeit with the eigenvalue $B(1 - \mathrm{e}^{-2\rho})$. We construct similar eigenvectors $g_{(u,v)} : X \to \mathbb{R}$ for a fixed $u \in W$ and general $v \in W \setminus \{u\}$. $g_{(u,v)}$ needs to satisfy the system of equations

$$\begin{cases} p(u,u)g(u) + p(u,v)g(v) &= \lambda g(u) \\ p(v,v)g(v) + p(v,u)g(u) &= \lambda g(v) \\ p(z,u)g(u) + p(z,v)g(v) &= 0 \end{cases} \ , \tag{2}$$

with $z \in X \setminus \{u, v\}$. Again, the final equation implies $g(u) = -g(v)$, but now this cause the top two equations to evaluate to

$$0 = \lambda g(u) \quad \text{and} \quad 0 = \lambda g(v) \ .$$

This can only be satisfied by $\lambda = 0$. Thus, putting $g_{(u,v)}(u) = 1$ gives us an eigenvector in the null space of P. By the analysis in Section 2.2, we know that there exists a M-th eigenvector that is not a linear combination of the eigenvectors found so far, but determining it analytically is hard. Luckily, using the trace of P, we can determine the corresponding eigenvalue λ_M, which is all we really need. We have that

$$\sum_{\ell=1}^{n} \lambda_\ell = tr(P) = \sum_{x \in X} \langle P 1_x, 1_x \rangle = \sum_{x \in X} p(x, x) \ ,$$

using the indicator functions $\{1_x\}_{x \in X}$ on X as the basis of $L^2(X, d\mu_c)$ for the matrix representation of P. This gives us that the final eigenvalue is given by

$$\lambda_M = \frac{1 + (N - 1)\mathrm{e}^{-2\rho}}{1 + (M - N)\mathrm{e}^{-\rho} + (N - 1)\mathrm{e}^{-2\rho}} - \frac{N\mathrm{e}^{-\rho}}{(M - N) + N\mathrm{e}^{-\rho}} \ .$$

Rewriting λ_M as a single fraction reduces the formula to

$$\lambda_M = \frac{(M - N)(1 - \mathrm{e}^{-2\rho})}{(1 + (M - N)\mathrm{e}^{-\rho} + (N - 1)\mathrm{e}^{-2\rho})((M - N) + N\mathrm{e}^{-\rho})} \ ,$$

which shows that λ_M is positive. Similarly, the difference $\lambda_M - B(1-\mathrm{e}^{-2\rho})$ expressed as a single fraction is given by

$$\lambda_M - B(1-\mathrm{e}^{-2\rho}) = \frac{-N\mathrm{e}^{-\rho}(1-\mathrm{e}^{-2\rho})}{(1+(M-N)\mathrm{e}^{-\rho}+(N-1)\mathrm{e}^{-2\rho})((M-N)+N\mathrm{e}^{-\rho})} ,$$

therefore $B(1-\mathrm{e}^{-2\rho}) > \lambda_M$, so λ_M is the smallest non-zero eigenvalue of P. This shows that the inclusion of unmarked images makes it a little easier to represent the diffusion distances in lower dimensional spaces. Instead of having $N-1$ equally large eigenvalues when $M = N$, one of them has become strictly smaller than the rest when $M > N$ and this difference increases if M is increased and N is kept fixed. This is in contrast to PCA, where the inclusion of more and more unmarked images increases the difficulty of embedding the data. Unfortunately, there are still $N-2$ equally important dimensions from which only two can be selected to represent the dataset.

To gain some insight into the way diffusion maps perform on the test environment e_1, we look at the dimensionality reduction of the first M images produced by putting $M = 7$ and $N = 5$. Figure 7 shows the images projected onto a two-dimensional space by a diffusion map with $R = 1$. Although the result is not as bad as the projection produced by PCA shown in Figure 4, the circular structure present in the images produced is not yet recovered.

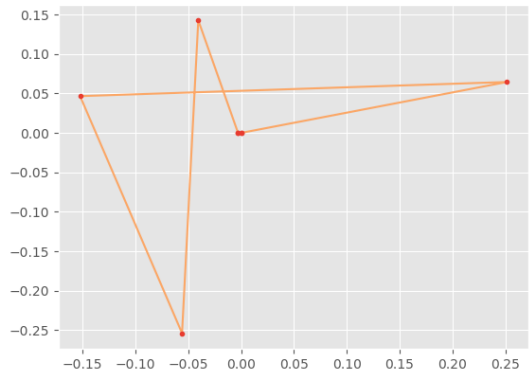

Figure 7: Two-dimesional embedding of the images produced by the test environment e_1 using a diffusion map.

3.3.4 Generalisation of the initial test

Diffusion maps and PCA both perform poorly on the test environment e_1 for the same reasons: the values that they use to order the dimensions in terms of importance show no clear preference. We believe that the performance of diffusion maps can be improved by increasing the width of the black marking in the environment. Currently when $M = N$, there is no difference in the distance between consecutive images and random images. Consequentially, this important structural component is not taken into account by the diffusion maps. Widening the black marker does emphasize this aspect. Consider images a_1, a_2 and a_3 given in vector notation by

$$a_1 = (1,1,0,0) , \quad a_2 = (0,1,1,0) \quad \text{and} \quad a_3 = (0,0,1,1) .$$

Using the 1-norm, the distance between a_1 and a_2 is smaller than the distance between a_1 and a_3:

$$||a_1 - a_2||_1 = 2 < 4 = ||a_1 - a_3||_1 .$$

This brings us to defining a generalisation of the initial test. Let $e_K : \mathbb{Z}/M\mathbb{Z} \to \{0, 1/G, \ldots, 1\}$ be the test environment given by

$$e_K([i]) = \begin{cases} 1, & \text{if } 0 \le i \le K-1 \\ 0, & \text{otherwise} \end{cases} ,$$

with $K \le M$ the size of the black marking. Again, we consider the M first images $\mathrm{im}_1, \ldots, \mathrm{im}_M$.

3.3.5 PCA's and diffusion maps' performance on the generalised test

Unfortunately, the widening of the black marking makes it difficult to analyse the methods by hand. Therefore, we rely on some specific examples to evaluate the performance of PCA and diffusion maps on the generalised test environments. For diffusion maps, we again put $\alpha = 0$ and $R = 1$.

First, we look at the embedding of images generated by the test environment e_3 with $M = 10$ and $N = 5$. Figure 8 shows the two-dimensional embeddings produced by PCA and diffusion maps.

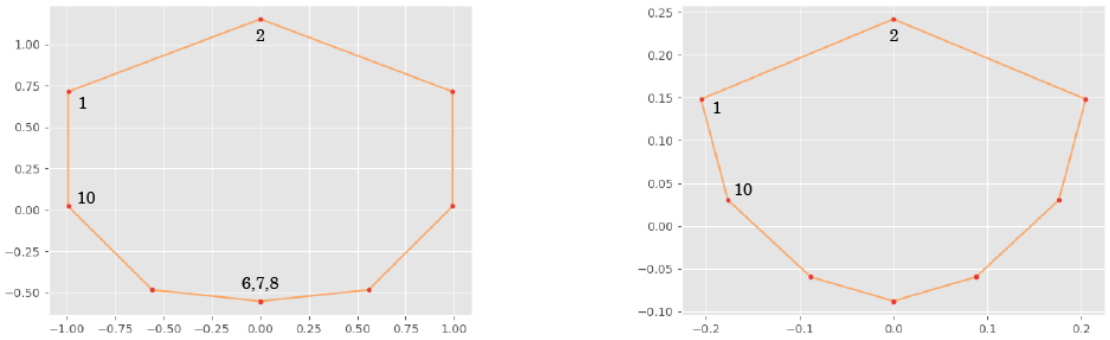

Figure 8: Two-dimesional embeddings of the images produced by the test environment e_3 with $M = 10$ and $N = 5$ using PCA (left) and a diffusion map (right).

Both graphs show a clear circular structure. Note that in both embeddings a larger distance between consecutive embedded images corresponds to a larger distance between the images themselves as exemplified by the images im_1, im_2 and im_{10}. The bottom of the embedding produced by a diffusion map seems rounder than that of the one produced by PCA, but the opposite is true for the top of the image.

We further investigate PCA by examining the principal components found by representing them as images after having undone the centring. The results are shown in Figure 9.

Figure 9: Representation of the principal components identified by PCA for test environment e_3 as images.

It seems like the combination of these components are better suited to representing multiple images, in contrast to the principal components identified for the images produced by the initial test environment e_1, which were over-fitted to represent the first image. This is also exemplified by the projections of the embedded images onto the two-dimensional linear subspace represented as images in Figure 10.

Image im_2 is clearly approximated by a positive rescaling of the second principal component, whereas the images im_6, im_7 and im_8 are approximated by negatively rescaling the second principal component. This does show that PCA is not able to properly reconstruct the original images, as the images im_6, im_7 and im_8 are completely blank, but their approximations show quite some variation in pixelvalues. This is partially exaggerated by the method used to represent general vectors in $\mathbb{R}^{\{1,\dots,N\}}$as images, since the approximation of these images in vector notation is given by

$$(0.26283707,\ 0.02074381,\ -0.08156266,\ 0.02074381,\ 0.26283707)\ .$$

Both PCA and diffusion maps seem to perform better on images with a widened marking as compared to those generated by the initial test environment. There appears to be variation in the singular values and eigenvalues used to discern important dimensions needed to represent the data.

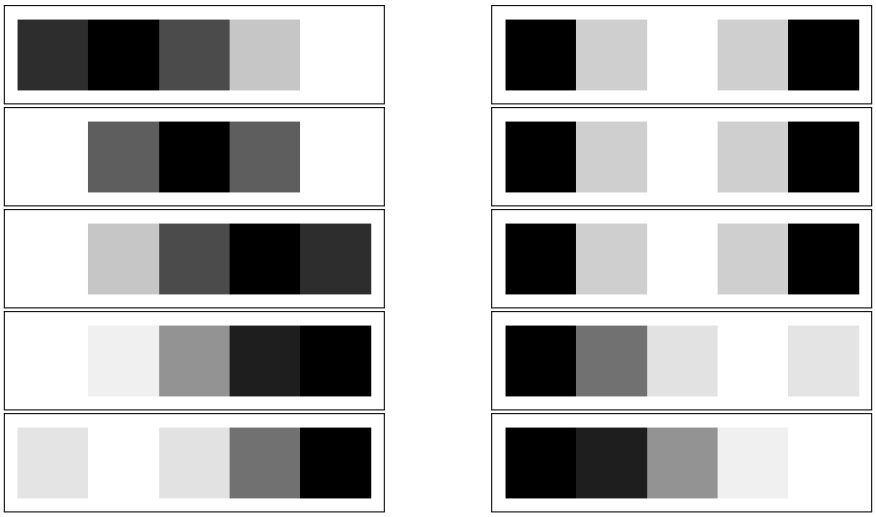

Figure 10: Representation of the projections of $\mathrm{im}_1, \ldots, \mathrm{im}_5$ (left, from top to bottom) and $\mathrm{im}_6, \ldots, \mathrm{im}_{10}$ (right) as images.

We now apply PCA and diffusion maps on images generated by a test environment that are reflective of images actually collected by Cozmo: its camera is 320 pixels wide, its field of vision is about 60 degrees in the horizontal direction and the smallest marker we were able to recreate in a real-world environment was eight pixels wide, see Section 4 and 5.1. This leaves us to consider the test environment e_8 with $M = 1920$ and $N = 320$. Figure 11 shows the embedding of the images produced by PCA and diffusion maps.

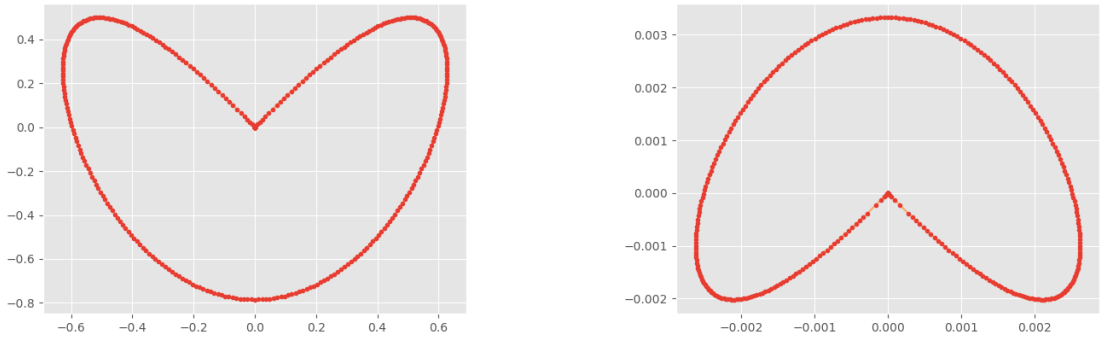

Figure 11: Two-dimesional embeddings of the images produced by the test environment e_8 with $M = 1920$ and $N = 320$ using PCA (left) and a diffusion map (right).

The embeddings look very similar, like the merging of two ellipses. Note that in both images, the unmarked images are mapped to the zero-vector at the center of the indentation. If not for this indentation, both embeddings would have succeeded nicely in uncovering the circular geometry underlying the images generated.

All-in-all, these results have given us good hopes for the performance of PCA and diffusion maps in real life environments. We think it unlikely that a pattern of only a single pixel wide is observed, since Cozmo's camera has a resolution of only 320×240 pixels, whilst having a field of vision of 58.40 degrees horizontally and 45.42 degrees vertically. Because of this, the methods ought to perform more like when applied to test environments e_K with $K > 1$ than like when applied to the initial test environment e_1.

4 Methodology

With this section starts the practical part of the report, describing the experiments performed using images collected by Cozmo. We start by detailing the environments and the movements performed by Cozmo, followed by specifying several image pre-processing methods intended to improve the quality of the lower-dimensional embeddings. Afterwards, the actual experiments are enumerated. These are separated into three categories, namely experiments used to

I) validate the simplified model from the Theoretical Exploration;

II) evaluate the performance of the dimensionality reduction techniques in practice and see how the resulting embeddings can be optimised;

III) evaluate the performance of the optimised dimensionality reduction techniques on images collected by having Cozmo perform more complex patterns of movement.

The images were collected with a Cozmo toy robot developed by Anki, Inc. It is equipped with a camera capable of capturing 320×240 pixel images with a frequency of 30 Hertz. By calibration of the lens in the factory, the position of the optical center of the projection within the images is determined to be 157.28 and 105.93 in floating point pixelvalues in the horizontal and vertical direction respectively. Furthermore, the focal length combined with pixel skew is 286.31 and 286.74 in floating point pixel values in horizontal and vertical direction. The camera's field of view is 58.40 and 45.42 degrees in horizontal and vertical direction. Cozmo is controlled by running a Python 3 script using Cozmo's SDK.

4.1 Environments

The experiments are performed in two environments, inside a cardboard cylinder and on a side table, both located in the authors' apartment. Both locations are described in more detail below.

4.1.1 Cardboard cylinder

The first category of experiments is aimed at validating the simplified model presented and used in the Theoretical Exploration. For the purpose of recreating the test environments, a cardboard cylinder was created which can be covered on the inside with sheets of paper showing specific patterns. If Cozmo is placed in the center of the cylinder with its head turned up by an angle of ten degrees with respect to the table's surface, the inside of the cylinder covers Cozmo's entire field of vision. The cylinder has a diameter of 22.3 centimetres and a height of 15.8 centimetres. For the experiments, the cylinder is placed on top of the dinner table in the living room (see Section 4.1.2) and illuminated by a single light source situated directly above the cylinder. To ensure that no outside light source influences this set-up, the experiments using the cardboard cylinder are performed after sunset. This set-up is shown in Figure 4.1.1.

4.1.2 Living Room

The second and third categories of experiments are performed with Cozmo on a side table in the living room. By default, Cozmo is placed in the center of the table, facing eastwards. The environment observed by Cozmo is the living room itself. The floor in the living room is covered with dark brown laminate, the walls painted white except for a single one painted red. Two windows allow daylight to pass into the living room, a small window in the northern wall and a large window overlooking the east-side of the apartment. The light entering the living room strongly influences its appearance to Cozmo, hence to keep the results consistent, all experiments are performed around noon. This allows the room to be illuminated by daylight, but prevents beams of direct sunlight from entering through the windows. The living room is starkly furnished, most noticeably are the couch near the side table and the dinner table in the middle of the room. The layout of the living room is shown in more detail in Figure 4.1.2. The

Figure 12: The cardboard cylinder with a test environment illuminated from above.

stark nature of the living room makes for a simple, real-life test environment for Cozmo. We remark that the surface of the side table Cozmo will be placed on reflects quite a lot of light.

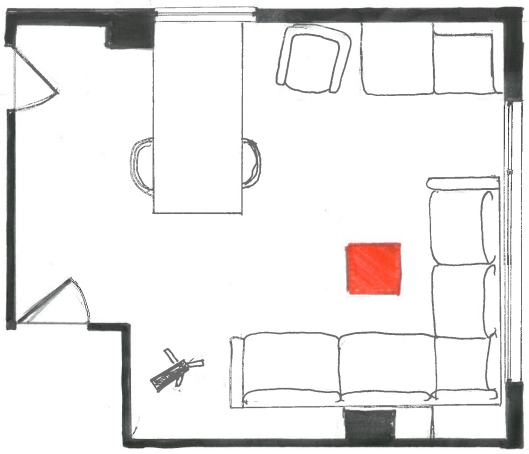

Figure 13: 1:100 scale map of the living room with the side table marked in red, drawn by Alessa Klomp.

4.2 Image collection

The first two sets of experiments will be performed with images collected by having Cozmo turn in place. The third set explores the ability of the dimensionality reduction methods to recover more complex patterns of movement. All movements assume that Cozmo is taken off his charger and with its lift turned all the way down. Cozmo's head is turned up by ten degrees with respect to the table's surface by default. The specifics of each movement are outlined in subsections below.

When the image collection process is started, images are collected automatically whenever a new raw camera images is available. This is achieved by listening to the `EvtRawCameraImage` event generated by Cozmo's software when a new image is available. After a specific number of images are collected, the image collection process is stopped.

4.2.1 Spinning in place

The topologically most interesting shape to try and recover in the experiments is the circle formed by spinning Cozmo in place. The motors controlling Cozmo's left and right treads are set to turn backwards and forwards respectively at a speed of 50 mm/sec, resulting in Cozmo spinning in a counter-clockwise direction. Having given Cozmo a second to accelerate, 240 images are collected whilst Cozmo remains spinning. This is sufficient for Cozmo to record (approximately) three full turns. After the images are collected, Cozmo's motors are stopped.

4.2.2 Spinning with a tilted head

A slightly more difficult set of images is generated by having Cozmo spin in place with its head tilted at different head angles. We let Cozmo spin in place as described in Section 4.2.1 five times with its head tilted by -25.00, -7.625, 9.75, 27.125 and 44.50 degrees respectively. A possible embedding of the images collected is five cirlces lying on top of each other.

4.2.3 Spinning at multiple locations

Another way to increase the difficulty of the image set is to have Cozmo spin in place at multiple locations. We let Cozmo spin as described in Section 4.2.1 at five places on the side table: near the four corners and in the center of the table. We hope that the dimensionality reduction techniques are able to uncover the different locations in addition to the spinning motion.

4.2.4 Driving around randomly

The two extensions presented above are very controlled. Cozmo's position can also be encoded into the structure of the images collected by having Cozmo drive around randomly on the table's surface. We start the image collection process and have Cozmo's motors controlling the left and right caterpillar tracks move at speeds given by the random variables $S + O$ and $S - O$ respectively, with $S \sim \text{Unif}(\{0, \ldots, 75\})$ representing Cozmo's forward movement speed in mm/sec and $O \sim \text{Unif}(\{-100, \ldots, 100\})$ the offset of the motors in mm/sec. Random speeds for the motors to move at, are generated twice a second. Having Cozmo drive around randomly has the disadvantage that it might drive off the table's edge. To prevent this, Cozmo's edge detection is enabled, stopping Cozmo just before it is about to fall off the table. After having stopped at an edge, Cozmo drives ten centimetres backwards and turns around clockwise or counter-clockwise at an angle of $A \sim \text{Unif}(\{90, \ldots, 180\})$ degrees before the motors are again instructed to move at random speeds. After a 1200 images have been collected, Cozmo is stopped.

There are three drawbacks to this collection method. Firstly, we were unable to make Cozmo stop to avoid objects, requiring any objects to be removed from the table this collection method is performed on. Secondly, when Cozmo approaches an edge at a narrow angle, it is still able to drive off the table despite the precautions taken. This issue was addressed by covering the floor underneath the side table with blankets and pillows to prevent Cozmo from being damaged and by performing the collection process repeatedly until the desired amount of images was collected. Finally, when Cozmo detects an edge, it performs a small animation, shaking its head up and down. We were unable to disable this animation, so Cozmo's head tilt needed to be reset to ten degrees each time after it had backed away from the edge.

4.3 Pre-processing images

Before applying the dimensionality reduction methods to the images collected by Cozmo, the images can be pre-processed such that the lower-dimensional embeddings are shaped more nicely. We examine the effects of blurring and renormalising the images. Additionally, instead of working with the images themselves, we investigate the effects of using the absolute difference between images as the input to PCA and diffusion maps.

4.3.1 Blurring

Motivated by the results from the Theoretical Exploration, it is possible that both dimensionality reduction methods perform poorly on environments with too fine details. Small details in images can be spread out by convolving the pixels in the images with a Gaussian kernel. The amount of blur varies with the standard deviation of the bell-curve expressed in amount of pixels.

4.3.2 Normalisation

Whilst working with the dimensionality reduction methods, we found out that the images were being sorted by their average pixelvalue, as shown visually in Figure 23. Wanting to make this structure appear less profound in the dataset, we chose to renormalise the images by dividing the vector representation of the images by their size according to a specific metric. For the choice of metrics, we restricted ourselves to the metrics used by the dimensionality reduction techniques themselves.

4.3.3 Difference between images

Edge detection is a commonly used technique to identify important features in an image. (Here, edge detection does not refer to Cozmo's ability to detect whether it is about to drive of an edge.) A cheap way to detect edges in a image is to look at the absolute difference between images. Moreover, looking at the difference between images allows us to identify jumps in overall brightness between consecutive images, caused by a beam of light previously not captured hitting the camera in a successive image. These jumps are also present in the lower-dimensional embeddings, so being able to remove them would certainly improve the embeddings, as they are not representative of the consistent speed Cozmo was moving at whilst collecting the images (excluding the random movement from Section 4.2.4).

4.4 Overview of the experiments

The first category of experiments is used to validate the simplified model from Section 3.2. To control the environment observed by Cozmo, the images are collected by having it spin in place inside the cardboard cylinder. The inside of the cylinder is covered with white sheets of paper and a single vertical black strip. The black strip is consecutively taken to be 1, 5 and 25 millimetres wide, creating three sets of images. From each of these sets of images, we derive another set of images by only considering the 160^{th} row of pixels from each image. We let both PCA and diffusion maps create a two-dimensional embedding of each image set for a total of twelve experiments. For the diffusion maps, we use the same parameters and metric used in the theoretical analysis, i.e. putting $\alpha = 0, R = 1$ and using the 1-norm as the metric.

The second category consists of experiments aimed at optimising the performance of the dimensionality reduction methods in real-life test environments. For these experiments, a single set of images was collected by spinning Cozmo in place in the center of the side table in the living room. Before testing the effects of pre-processing or different parameter choices, a baseline is created by having both PCA and diffusion maps produce a two-dimensional embedding of the raw image set. By default, the parameters for the diffusion maps are set to $\alpha = 1$ and $R = 1$ with the 1-norm as the default metric. Blurring the images is tested with the bell-curve having a standard deviation of $0.1, 3, 10$ and 100. Next, the images are normalised consecutively by the 1-norm, 2-norm, 5-norm, 10-norm and the sup-norm. PCA is only applied to the images normalised with respect to the 2-norm, whereas diffusion maps are applied to all normalised images using the same metric internally as used to normalise the images. Thirdly, we look at the absolute difference between images, with sudden jumps in brightness filtered out manually. PCA and diffusion maps are applied to these images and their normalisation with respect to the 2-norm. Concerning parameter choices for diffusion maps, α is consecutively put to $0, 0.5$ and 1 and the rescaled average distance R is put to $0.04, 0.2, 1, 5, 25$ and 50. The effects of altering the parameters are tested on the images normalised with respect to the 2-norm which gave the best results of all pre-processing techniques.

Last but not least are the experiments used to evaluate the performance of the optimised dimensionality

reduction techniques on more complex image sets. Three sets of images are collected. The first is collected by spinning Cozmo in place with its head tilted at several angles in the center of the side table. The second is collected by spinning Cozmo in place at several positions on the side table and the third image set is collected by having Cozmo drive randomly across the side table's surface. The best dimensionality reduction technique identified by the experiments in the previous category turns out to be applying diffusion maps to image sets normalised with respect to the 2-norm and with the parameters $\alpha = 1$ and $R = 0.2$. This method is used to create a three-dimensional embedding of the first image set and four-dimensional embeddings of the second and third image sets.

5 Results and discussion

Time to see how PCA and diffusion maps perform in practice! Here, we describe and discuss the results of the experiments described in the Methodology. To do so, we utilise a large number of two-dimensional embeddings of collected images. The first coordinate of the embedding is always plotted along the horizontal axis and the second coordinate along the vertical axis. We have taken the liberty of flipping embeddings horizontally, vertically or both to make it easier to compare between them.

5.1 Validating the simplified model

Our first goal is to check if our simplified model from Section 3.2 is representative of reality. Figure 14 shows some images captured by Cozmo from the inside of the cardboard cylinder whose decoration was an attempt to recreate the test environments from the Theoretical Exploration in practice. One immediately notices that the white background is observed as a dark shade of grey and that the pixels at the edges are darker than the ones in the middle. The latter points to a structural deviation in Cozmo's camera, which was unaccounted for in the simplified model. We remark that we were unable to create the one-pixel-wide black marker from the Theoretical Exploration, as even the smallest black strip of one millimetre wide is represented by eight pixels in the camera image.

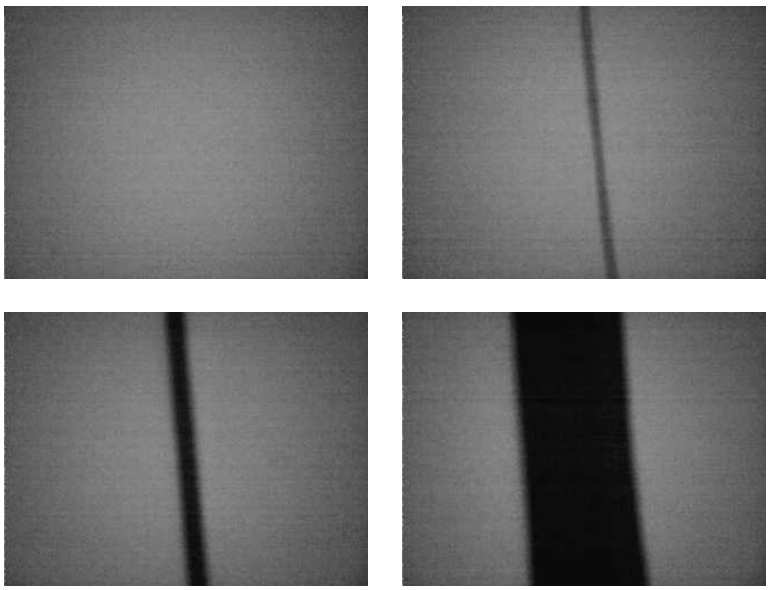

Figure 14: Images captured by Cozmo showing no black strip, a black strip of 1, 5 and 25 millimetres.

Figures 15 and 16 show the results of applying PCA and diffusion maps respectively to the images collected from inside the cylinder for the varying widths of the black stripes. The simplified model assumes that it suffices to consider only images of a single pixel high, since Cozmo's camera only moves in the horizontal direction. For this reason, the aforementioned figures also show the embeddings produced when only considering the 160^{th} row of pixels from every image. The rows show a similar structure as the full images, except for the for the embedding produced by PCA of the black strip of five millimetres, where the full images show a clearer 'loopy' structure.

For both PCA and diffusion maps, only the embeddings corresponding to a strip of 25 millimetres show a circular shape, whilst the simplified model predicted that even the embeddings with a strip of 1 millimetre would show a clear structure. The shape of the embeddings with a strip of 25 millimetres is also slightly different than that of the embeddings in Figure 11: the unmarked images are pulled outwards instead of creating an indentation. This might be caused by the marker being wider, since the embeddings are more similar to those in Figure 8. The other embeddings probably show so little structure because of the amount of noise in the images, overwhelming the structure from the black marking. One could also

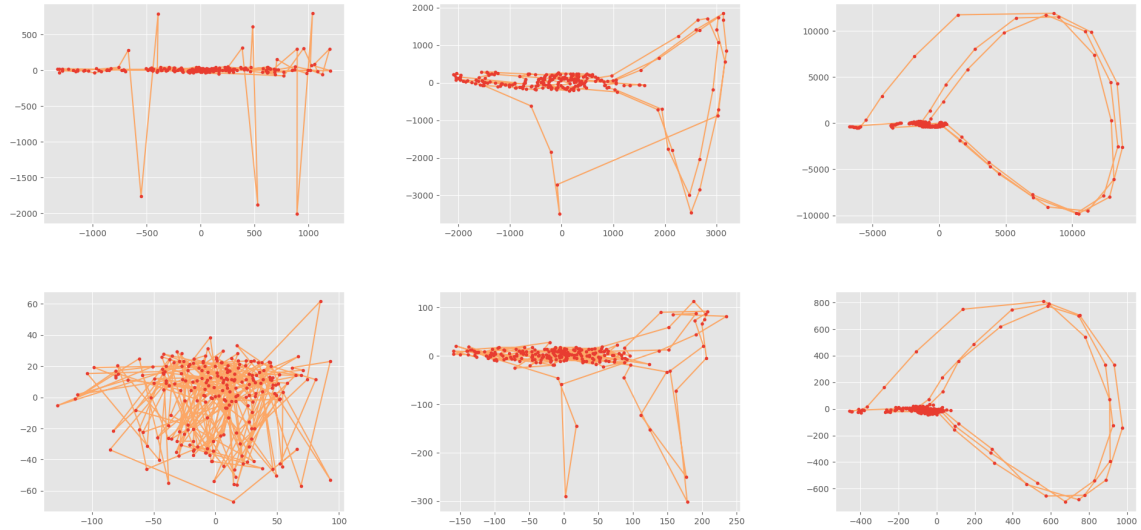

Figure 15: Two-dimensional embeddings of the images (top) and their 160$^{\text{th}}$ row of pixels (bottom) from the cylinders with a black strip of 1, 5 and 25 millimetres in width (left to right) produced by PCA.

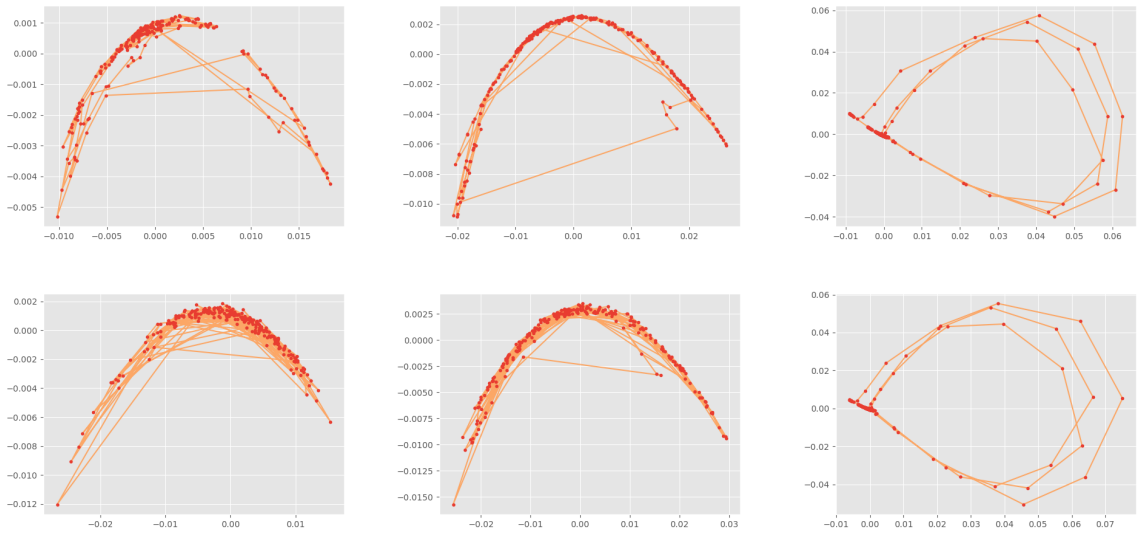

Figure 16: Two-dimensional embeddings of the images (top) and their 160$^{\text{th}}$ row of pixels (bottom) from the cylinders with a black strip of 1, 5 and 25 mm's in width (left to right) produced by diffusion maps.

suspect the background being much darker or the pixels near the edges being darker than those in the center, but these attributes would have been filtered out by the centring of the images by PCA.

Considering the results above, the simplified model does not seem to represent reality all to well. However, it predicted that PCA and diffusion maps could uncover the underlying geometry of the images for specific environments, which they did. But how well do they perform in real-life environments?

5.2 Optimisation

Figure 17 shows the basic embeddings produced by PCA and diffusion maps if Cozmo is spun around in place. Unfortunately, these do not reflect the circular structure of the movement performed by Cozmo when collecting the images. PCA's embedding looks like a helix, but it is plagued by sudden large jumps,

caused by a sharp change in overall brightness between consecutive images as exemplified by the images in Figure 18. The embedding produced by the diffusion map looks even less circular. It crosses itself multiple times, resulting in a bird-like shape.

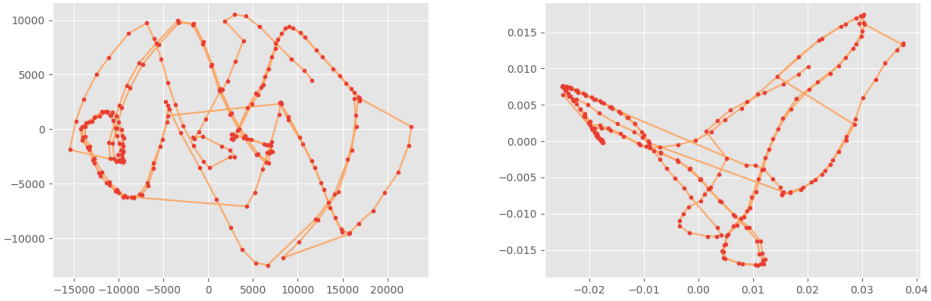

Figure 17: Two-dimensional embeddings of the images collected by spinning Cozmo in place produced by PCA (left) and diffusion maps (right).

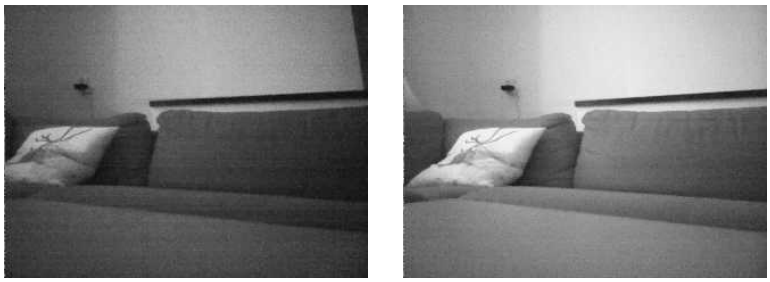

Figure 18: Consecutively collected images displaying a large difference in overall brightness.

5.2.1 Blurring

Luckily there are several options for pre-processing images and parameter choices to try and improve the quality of the embeddings. We start off by looking at the effects of blurring the images. As it is hard to asses the amount of blur based on numbers alone, Figure 19 shows the same image being blurred with increasing intensity until it has gotten completely unrecognisable in the end.

Figures 20 and 21 show the embeddings of the blurred images produced by PCA and diffusion maps respectively. The embedding produced by PCA seems unaffected by the first three types of blur. These blurrings only slightly change the embeddings produced by diffusion maps, note the swelling of the loop in the upper-right corner. When the images are blurred with a standard deviation of 100 however, the embeddings are clearly different. For PCA, it is unclear if it is better or worse than the original embedding. The left side is stretched out, whilst the right side is compressed. The embedding produced by the diffusion map has definitely degraded, reducing the shape to a one-dimensional structure.

All-in-all, blurring does not seem to improve the quality of the embeddings. Examining the blurred images themselves also shows that a standard deviation of 100 is too high, since it has become very hard to trace back the original image belonging to a blurred image. The images blurred with a standard deviation of ten already are quite blobby, showing no fine details whatsoever. Because of this, we judge any blur with a standard deviation higher than ten to be unusable. Since the blurs with a standard deviation lower than ten do not (significantly) effect the embeddings produced, we conclude that it is best not to blur the images at all.

We are left to wonder why blurring did not effect the embeddings unless taken to the extreme, as we expected that it would, after the Theoretical Exploration showed us that fine details are capable of throwing off the entire embedding. Looking at the principal components identified by PCA can offer us

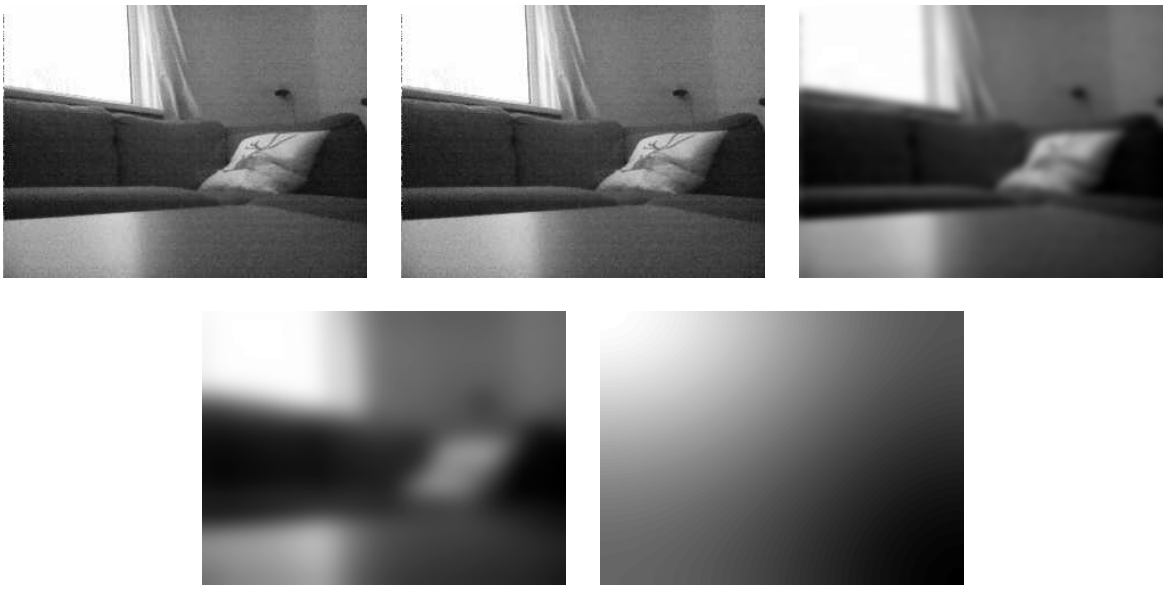

Figure 19: Image captured by Cozmo with no blur and blurred with a standard deviation of 0.1, 3, 10 and 100 (left to right, top to bottom).

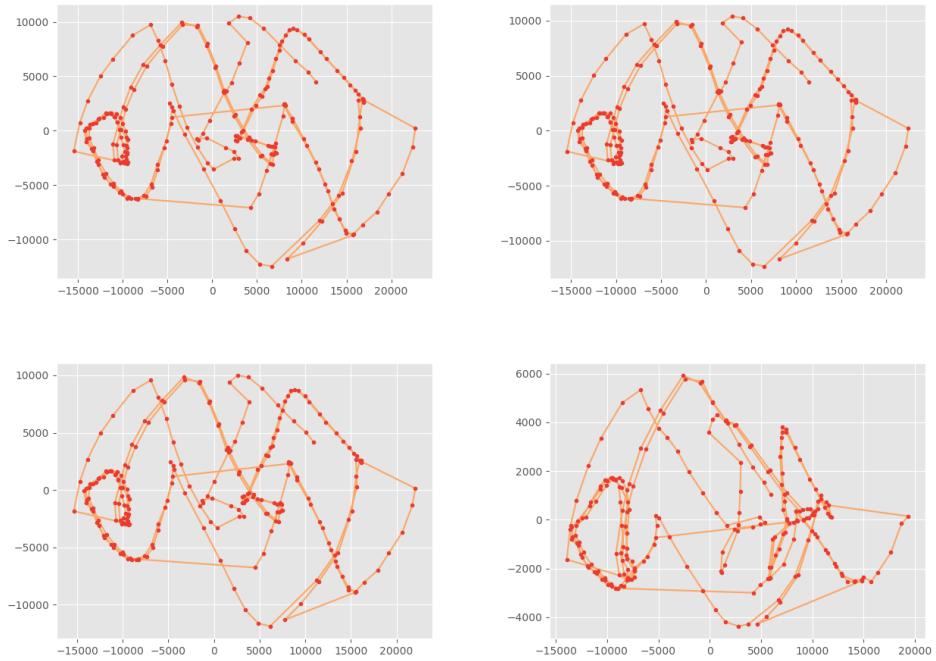

Figure 20: Two-dimensional embeddings of the images blurred with a standard deviation of 0.1, 3, 10 and 100 (left to right, top to bottom) produced by PCA.

some insight. Since only two principal components are used to approximate every image (after centring) we expect these components to already be quite blurry, as any fine detail would have to be present in a large number of images for it to be present in the principal components. Hence, removing details from the original images would have no effect on the principal components. Consequentially, the two-dimensional embeddings would also not be effected as they are linear combinations of the principal components. Figure 22 shows the principal components from the original images next to the components from the images blurred with a standard deviation of 10 and 100.

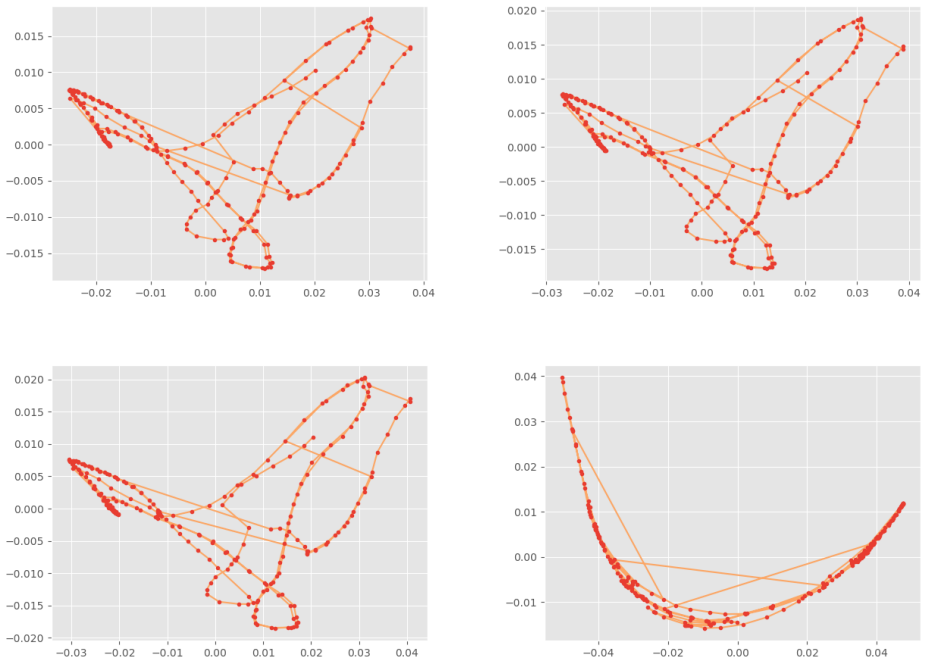

Figure 21: Two-dimensional embeddings of the images blurred with a standard deviation of 0.1, 3, 10 and 100 (left to right, top to bottom) produced by diffusion maps.

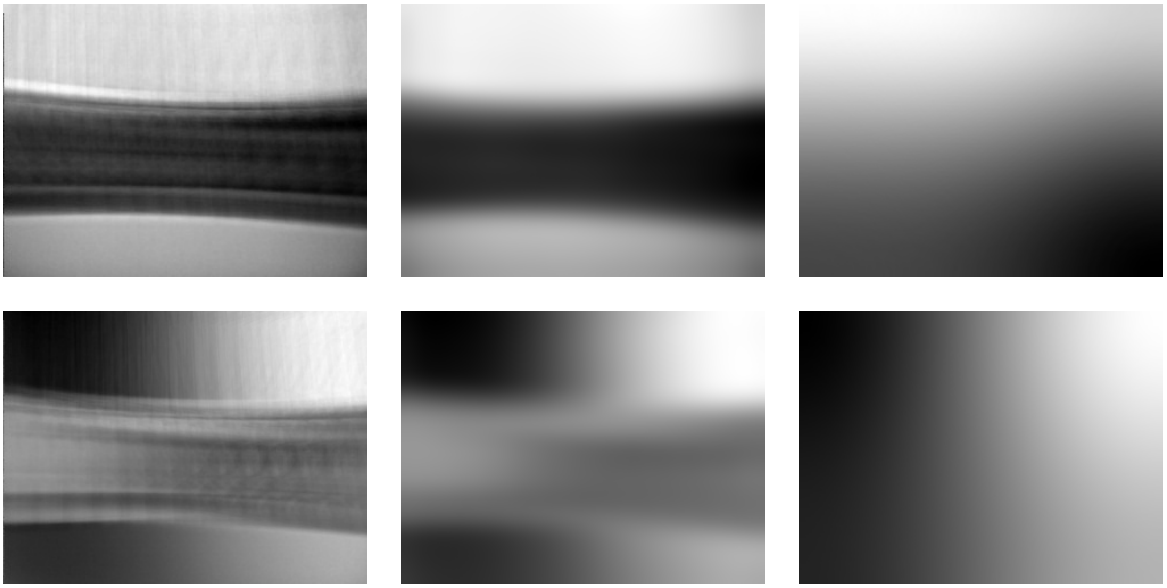

Figure 22: Principal components (top and bottom) identified by PCA for the images with no blur and blurred with a standard deviation of 10 and 100 (left to right).

Although the principal components of the original images are not as blurry as the principal components of the images blurred with a standard deviation of ten, they are quite similar. If one were to manually blur the original principal components by squinting one's eyes and looking through one's eyelashes, the principal components of the two sets seem even more alike. The images blurred with a standard deviation of 100 however are fundamentally different from the other principal components, so our suspicion seems to hold.

5.2.2 Normalisation

Next, we investigate the effects of normalising the images with respect to different metrics. Colouring the embedded images in relation to their average pixelvalue shows that this is the main structure discovered by PCA and diffusion maps, as shown in Figure 23. Because this structure is not of interest to us, we reduce its effects by normalising every image with respect to a specific metric, since normalising with respect to the 1-norm results in the average pixelvalue being the same for every image. Figure 23 also shows that after normalisation, the average pixelvalue indeed does no longer explain the structure of the embedding.

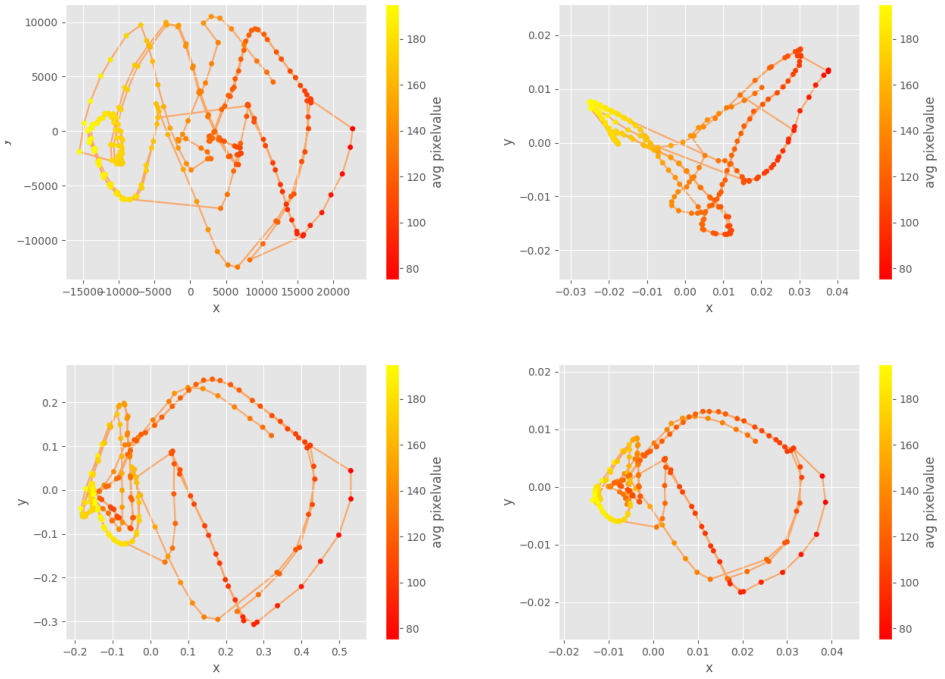

Figure 23: Two-dimensional embeddings of the images (top) and the images normalised with respect to the 2-norm (bottom) produced by PCA (left) and diffusion maps (right) with the embedded images coloured according to their average pixelvalue.

Figure 24 shows the effects of normalisation for PCA. The embedding has gotten more circular, but it still loops in on itself. Moreover, the images are distributed less uniformly over the geometry than before, clomping together a large number of images in a spot in the left-side of the diagram and the visualisation obscures the underlying geometry in that area.

A larger number of different metrics are tested when working with diffusion maps, the results are shown in Figure 25. As the parameter p of the p-norm increases, the embeddings get less round and less smooth, but the images are spread more uniformly over the geometry. Most noticeably is the loop on the left side which was not discernible for the 1-norm, but that is increasingly stretched out. Note also the loop that looks like an X whose bottom legs are connected, which grows and moves from the left to the right side of the plot.

For both PCA and diffusion maps, the embeddings of the normalised images better reflect the circular motion used to collect the images, so we want to employ this technique. Because of the trade-off between a smooth embedding and the heaping together of images however, there is no obvious choice for which metric to use. For any further experiments, we opt for using the 2-norm as it partially possesses both characteristics.

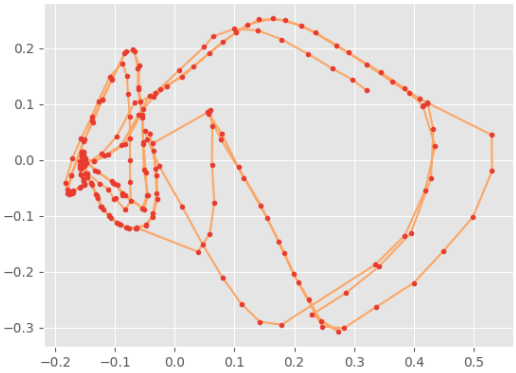

Figure 24: Two-dimensional embedding of the images normalised with respect to the 2-norm by PCA.

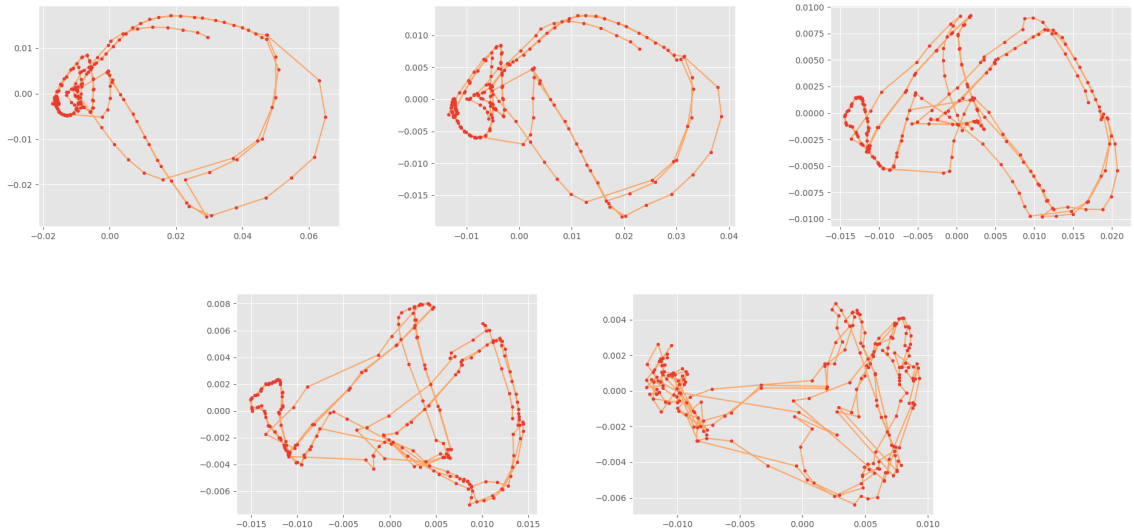

Figure 25: Two-dimensional embeddings of the images normalised with respect to the 1-, 2-, 5-, 10-, and sup-norm (left to right, top to bottom) by diffusion maps.

5.2.3 Difference between images

The final pre-processing technique we explore is applying the dimensionality reduction techniques to the absolute difference between images. After the computation of these images, five were filtered out, corresponding to the large jumps in overall brightness as seen in Figure 17. As Figure 26 shows, the images belonging to these jumps can easily be distinguished.

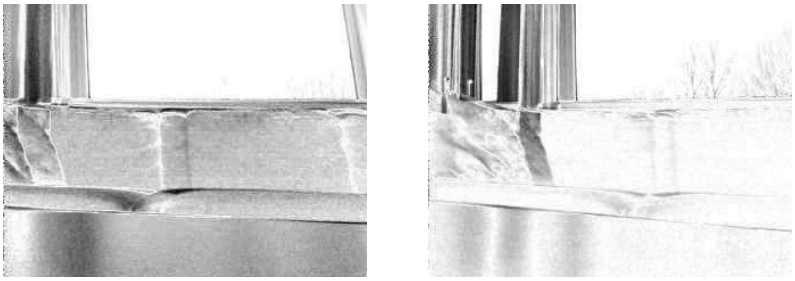

Figure 26: The difference between images with (left) and without (right) an overall jump in brightness.

Figure 27 shows the embeddings produced when applying PCA and diffusion maps to the differences between images. Note that the embeddings are very similar, as if the same three-dimensional object was viewed from slightly different angles. Additionally, the PCA's embedding looks nicer than the one produced by the diffusion map. If we normalise the differences with respect to the 2-norm, the embeddings produced by PCA and diffusion map look extremely similar. The embeddings also show a very circular structure, but going around multiple times per rotation by Cozmo.

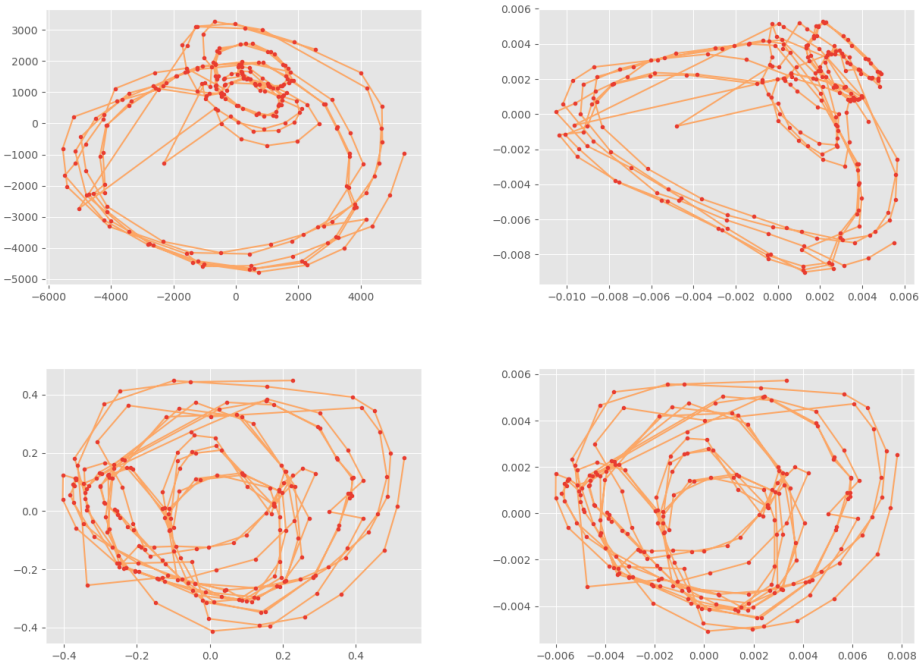

Figure 27: Two-dimensional embeddings of the difference between images (top) and the differences normalised with respect to the 2-norm (bottom) produced by PCA (left) and diffusion maps (right).

In an attempt to understand why the embedding winds up around itself, we examine the principal components (see 28) of the normalised differences. The principal components seem to only represent a contrast between light and dark pixels in the upper part of the image. These patterns resemble the principal components from the images generated by a test environment in Section 3.3.5, see Figure 9. These principal components were encoded the rotation of a single marking. Hence, the principal components of the difference between images probably encode the same images, meaning that every loop in the embedding corresponds to a marking in the top half of the image that moves from the left to the right side of the image. Inspecting several images supports this claim. Figure 29 shows three images taken from the bottom of multiple loops. All images show a black marker in the middle of the image's upper half.

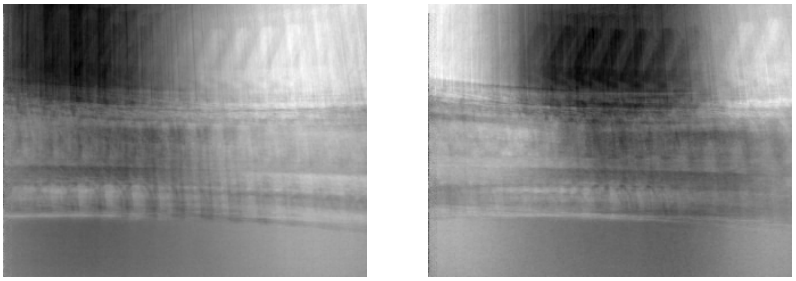

Figure 28: Principal components identified by PCA for normalised differences between images.

The embeddings of the absolute difference between consecutive images have a pleasant, circular shape, but unfortunately, taking differences also removes the information on to the global position of an image

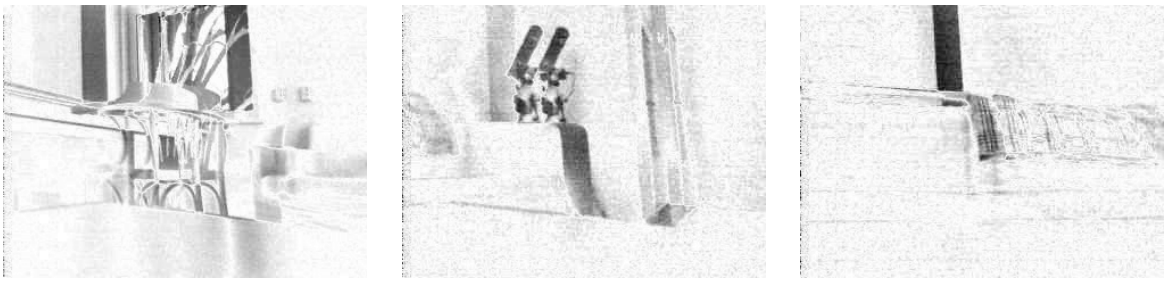

Figure 29: Three differences between images sampled from the bottom of loops in the left diagram in Figure 27.

within a single rotation performed by Cozmo, which is precisely the information we wish to uncover. Because of this, we stick to just looking at the images themselves. The ability to filter out jumps in overall brightness is useful though.

Having looked at all kinds of image pre-processing, we investigate the effects of multiple parameter choices for diffusion maps. Figure 30 shows the embeddings produced by putting α to 0, 0.5 and 1 consecutively. The embeddings hardly show any difference, so we continue to use the standard setting $\alpha = 1$.

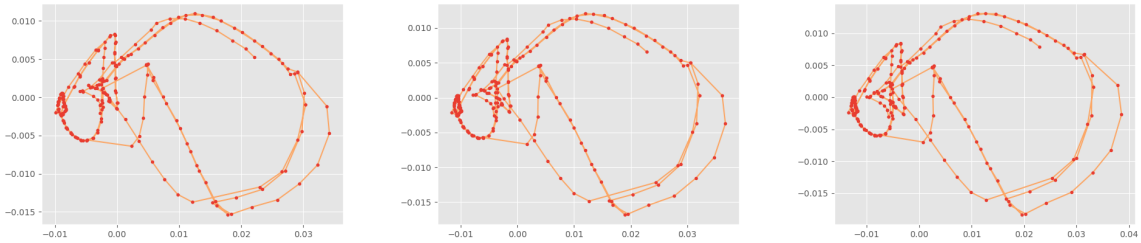

Figure 30: Two-dimensional embeddings of the images produced by diffusion maps with the parameter α put to 0, 0.5 and 1 (left to right).

Subsequently, we test several values for the rescaled average distance R. Figure 31 shows the embeddings produced. The second to fourth images show that when long distance relations are devalued, the loops on the left are contracted and the main loop is enlarged. This process is continued in the in the fifth diagram, with the loops previously on the left hand side all reduced to the appendix to the main loop in the lower-right corner. When we zoom in even more on the local geometry, even the main loop is partially mapped onto the same point as the images previously in the loops on the left side. The only images that stand out are those depicting the large and the small windows in the living room, the left and right 'legs' in the sixth diagram respectively, the images at the 'feet' are shown in Figure 32.

The embeddings above show that when the rescaled average distance is taken too large, a large part of the structure present is thrown away. It is easy to mistake such an embedding for a successful one. If you were to consider only the fifth embedding in Figure 31, it looks like the circular structure is recovered very nicely, but a large subset of images are mapped onto the same points. We wish not to hide such structures and to pull apart images that are being mapped onto the same area, therefore we consider $R = 0.2$ the best choice for the rescaled average distance between images.

With the parameters tuned, diffusion maps produce better embeddings than PCA. Because of this, we will only use diffusion maps with $\alpha = 1$ and $R = 0.2$ on images normalised with respect to the 2-norm for the experiments to come.

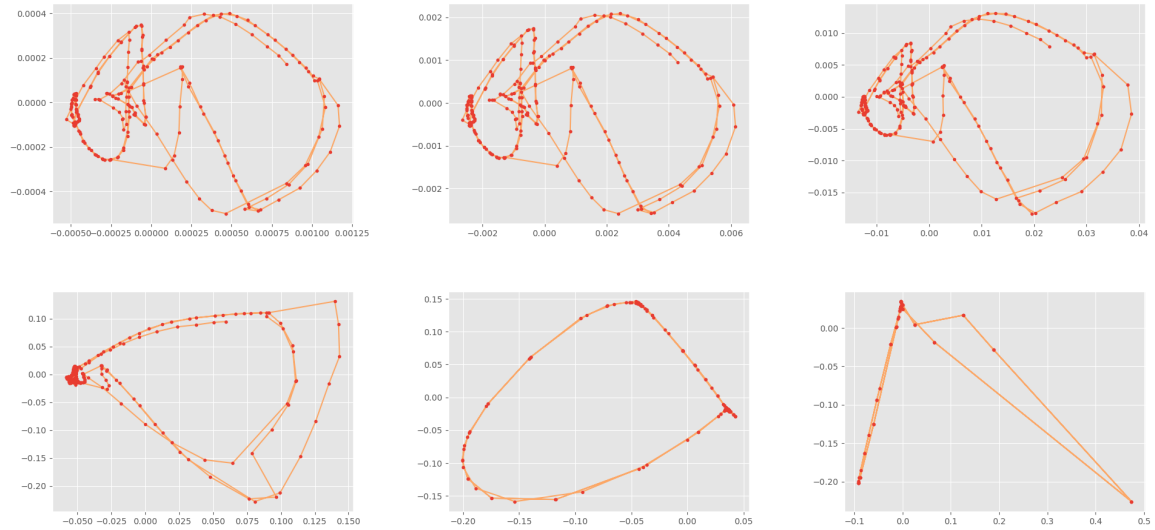

Figure 31: Two-dimensional embeddings of the images produced by diffusion maps with the parameter R put to 0.04, 0.2, 1, 5, 25 and 50 (left to right, top to bottom).

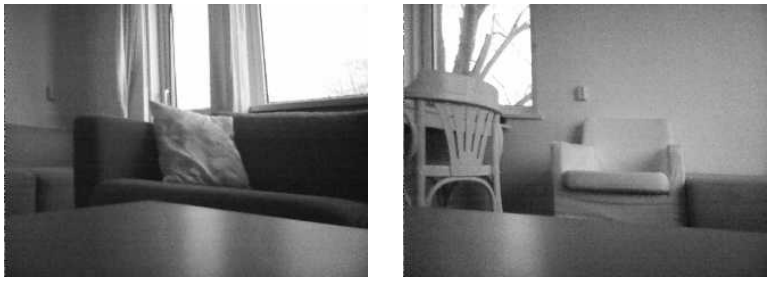

Figure 32: The images mapped onto the bottom of the 'legs' in the final embedding in Figure31.

5.3 More complex structures

We further test the performance of diffusion maps on images collected by having Cozmo move in more complex patterns.

5.3.1 Spinning with a tilted head

Figure 33 shows three, two-dimensional projections of the three-dimensional embedding of images collected by spinning Cozmo around with its head tilted at different angles. Without looking at colour, we are able to distinguish five loops, the red and blue loops overlapping too much to be easily separable. The five loops resemble the petals of an opened flower, overlapping at the floral bud. The images in the overlapping area depict the west-side of the living room, whereas the images in the opposing ends of the petals are of the large window on the east-side. This is similar to the embeddings of normalised images in the previous section, with images of the west-side being contracted. When looking at the embedding from a different viewpoint (see Figure 34), it is clear that the loops are not ordered neatly according to the tilting of the head. Consecutive loops seem to be rotated about by such an angle of approximately 120 degrees.

Despite the sometimes not very circularly shaped loops and the lumping together of a large number of images like in the embeddings from Section 5.2, the embedding is rather successful.

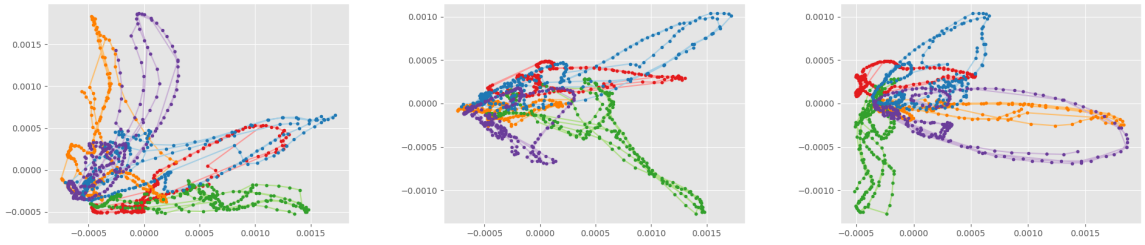

Figure 33: Top, front and side view (left to right) of the three-dimensional embedding of the images collected by spinning Cozmo in place with its head tilted at -25.00 (red), -7.625 (orange), 9.75 (green), 27.125 (blue) and 44.50 (purple) degrees.

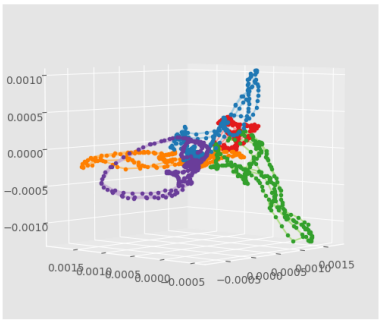

Figure 34: Rotated view of the three-dimensional embedding of the images collected by spinning Cozmo in place with its head tilted at -25.00 (red), -7.625 (orange), 9.75 (green), 27.125 (blue) and 44.50 (purple) degrees.

5.3.2 Spinning at multiple locations

The second experiment in this category exists of spinning Cozmo in five places on the side table in the living room. Figure 35 shows two important two-dimensional projections of the four-dimensional embedding produced by a diffusion map. The first projection shows a structure that can be traced back to Cozmo's position and the second one seems to represent the same rotation performed at every location, since it shows images from every location (i.e. colour in the diagram) tracing out the same circular geometry. The first projection is believed to encode position as in certain parts of the plot only two colours trace out the same shape with the colours sharing a positional attribute. For example, the blue and orange loops overlap in the upper-right corner and both sets of images were created by placing Cozmo near the northern edge of the side-table. The same is exemplified by the overlapping of only the blue and green loops in the upper-right corner which both were collected near the east-side of the table. Moreover, the overlapping parts show that the camera was pointed at the shared cardinal direction of the loops' positions, see Figure 36. The figure also features an image slightly left and down from the diagram's center where only the purple and green embeddings overlap. We were unable to identify a part of the diagram where only the orange and purple loops overlap, most likely because images portraying the east-side of the living room are all mapped very closed to one another.

All-in-all, the embedding is very successful, in particular the rather clean separation of Cozmo's position and rotation into the two different projections.

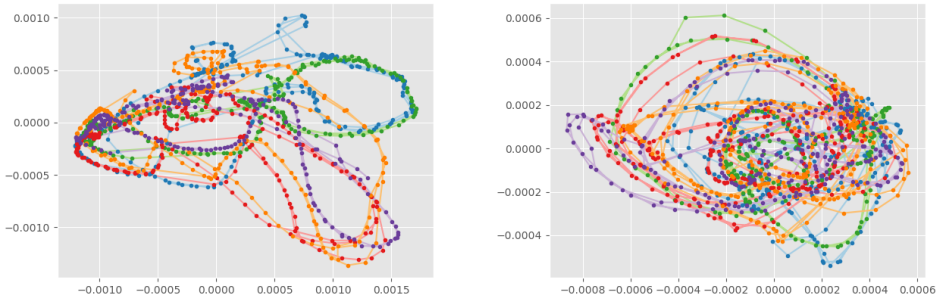

Figure 35: The first and second coordinates (left) and the third and fourth coordinates (right) from a four-dimensional embedding of the images collected by spinning Cozmo in multiple places, namely the north-east (blue), south-east (green), center (red), north-west (orange) and south-west (purple) corner of the side table.

Figure 36: Views shared by images in the overlap between the blue and orange, blue and green, and green and purple loops in Figure 35.

5.3.3 Driving randomly

Finally, we consider the four-dimensional embedding produced by letting Cozmo drive around randomly on the table's surface. Again, our aim is to recover Comzo's position and rotation during its random walk from separate views on the embedding. Figure 37 was created to compare the embedding against. It is a reconstruction of the path traversed by Cozmo based on the images collected.

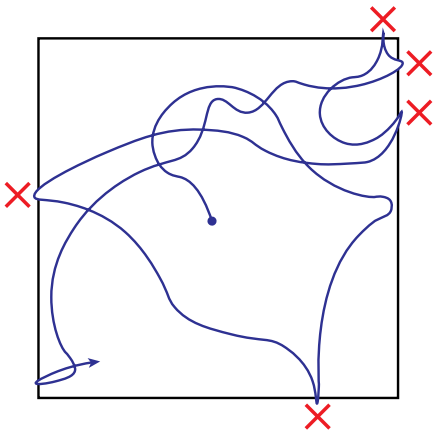

Figure 37: Reconstruction of Cozmo's random walk across the side table's surface with the red crosses indicating moments where Cozmo was about to drive off the table.

The disorderly nature of the path makes it difficult to find two-dimensional projections that convincingly

encode Cozmo's position or rotation, but we did find some strong contenders shown in Figure 38 and 39.

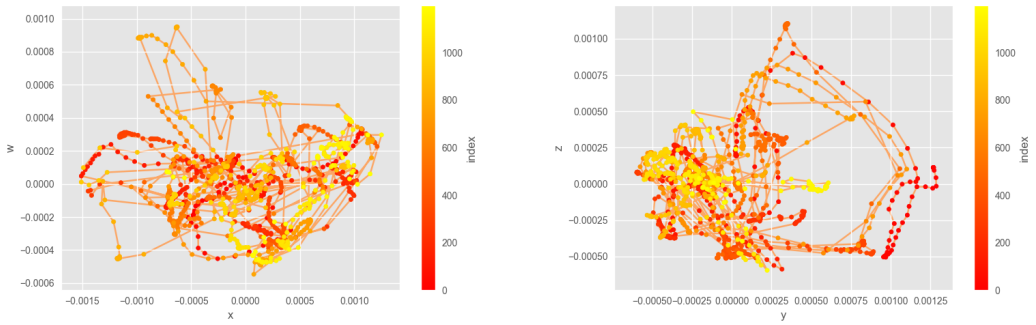

Figure 38: The second and third coordinates (left) and the first and fourth coordinates (right) from a four-dimensional embedding of the images collected by letting Cozmo drive around randomly with the embedded images coloured according to the order they were collected.

The first projection in Figure 38 seems to represent Cozmo's rotation, at the start at least. Initially, Cozmo's camera is pointed to the east, turning north before turning east again and eventually turning toward the south and the west. This corresponds to the embedding of the first couple of images in the first projection, with images looking north, east, south and west mapped to the bottom, right, top and left of the diagram respectively. As the first projection seems to encode Cozmo's rotation, we hope that the second projection resembles Cozmo's position. Sadly, the second projection is too cluttered to be able to recognise any patterns. The left part of the first projection suffers from the same problem.

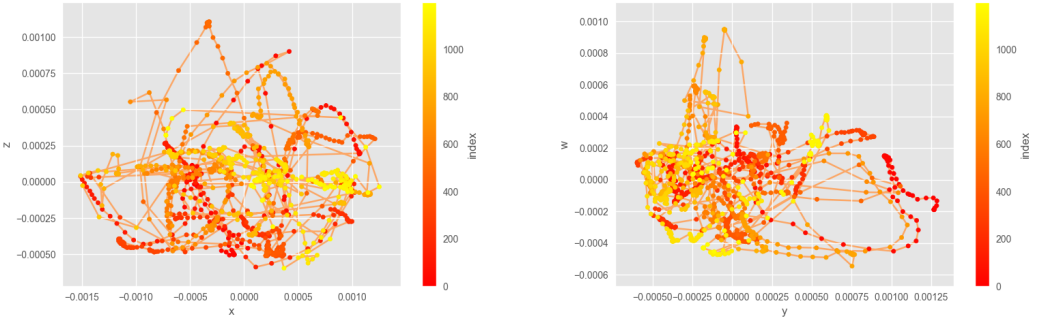

Figure 39: The first and third coordinates (left) and the second and fourth coordinates (right) from a four-dimensional embedding of the images collected by letting Cozmo drive around randomly with the embedded images coloured according to the order they were collected.

Let us have a look at the projections presented in Figure 39. The first projection faintly exhibits traces of Cozmo's rotation, showing curved trajectories e.g. the three curved 'hills' in the upper half of the diagram. Examining the images in these segments indeed show Cozmo's camera turning. There are however also a large number of embedded images that do not follow this pattern. The slightly descending segment just below the left 'hill' for example, which shows Cozmo driving towards the table's edge without moving its camera. Moreover, the horizontal axes in the second projection does not seem to represent a positional coordinate at all, since the embedded images start in the right-most part of the graph, whereas Cozmo started in the center of the table and Figure 37 shows that Cozmo drove past the center in every direction possible.

In conclusion, this embedding leaves a lot to be desired. Its images are often embedded too close to each other to be able to discern a global pattern.

6 Conclusion

Using PCA and the method of diffusion maps, we were reasonably able to uncover the geometries traced out during the image collection process. When Cozmo is spun in place, the embedding of the normalised images is pretty circular, but a lot of images are clomped together, obscuring the underlying structure. The same is true for when Cozmo is spun in place with its headed tilted at multiple angles. The images belonging to different head tilts are discernible, but they overlap quite a lot in segments where the images of a single head tilt are lumped together as well. Uncovering the position from images when Cozmo was spun in place at multiple locations worked well, three of the four cardinal directions were identified. When Cozmo is allowed to drive around randomly however, we were unable to find a clear representation of its position or rotation.

Although diffusion maps produced the better embeddings, the advantage of their non-linearity over the linear PCA is not very outspoken. The embeddings of the theoretical images were very similar, but diffusion maps produced a slightly cleaner embedding on the recreated test environment with a black strip of 25 millimetres. The embeddings of images normalised with respect to the 2-norm on their own also show no clear difference between the two methods, it were the parameters that allowed us to customise the diffusion maps that gave it the upper hand. PCA however offers us more insight into its inner workings by being able to represent the principal components as images, something we would like to have had for diffusion maps.

The main issue with the embeddings is the clomping together of images, hence future research could drastically improve their quality if a method could be found to mitigate this behaviour. Another, perhaps easier, approach would be to filter out the large differences between images that experience a jump in overall brightness. Trying to find a metric on the absolute difference between images might be a good starting point.

In conclusion, the dimensionality reduction techniques are relatively successful in uncovering the latent variables, but to properly recognise them in embeddings of more complex datasets does require prior knowledge of these variables. Thus, there is still work to be done before dimensionality reduction methods will provide useful insights into data on their own.

7 References

[1] Coifman, R.R. Lafon, S. (2006). Diffusion maps. *Applied and Computational Harmonic Analysis, 21*(1). 5-30.

[2] Chen, V. (2016, January 23). SVD and low-rank approximation. Retrieved from www.victorchen.org/2016/01/23/svd-and-low-rank-approximation .

[3] Grimmett, G. Stirzaker, D. (2001). *Probability and Random Processes* (3rd ed.). Oxford, Oxford University Press.

[4] Lafon, S.S. (2004). *Diffusion Maps and Geometric Harmonics*(PhD). Yale University.

[5] Lafon, S. Keller, Y. Coifman, R.R. (2006). Data Fusion and Multicue Data Matching by Diffusion Maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 28*(11). 1784-1797.

[6] Lee, J.A. Verleysen, M. (2007). *Nonlinear Dimensionality Reduction*. New York: Springer Science + Business Media.

[7] Lee, J.A., Verleysen, M. (2009). Quality assessment of dimensionality reduction: Rank-based criteria. *Neurocomputing, 72*(7-9). 1431-1443.

[8] Van der Maarten, L. Postma, E. Van den Herik, J. (2009). *Dimensionality Reduction: A Comparative Review* (TiCC TR 2009–005). Tilburg, Tilburg University.

[9] Martins, R.M. Coimbra, D.B. Minghim, Rosane. Telea, A.C. (2014). *Computers & Graphics, 41*. 26-42.

[10] Mokbel, B. Lueks, W. Gisbrecht, A. Hammer, B. (2013). Visualizing the quality of dimensionality reduction. *Neurocomputing, 112*. 109-123.

[11] Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *Philosophical Magazine, 2*: 559-572.