Learning A Geometry Integrated Image Appearance Manifold From A Small Training Set

Yilei Xu, Amit K. Roy-Chowdhury [*]
Department of Electrical Engineering
University of California, Riverside

Abstract

While low-dimensional image representations have been very popular in computer vision, they suffer from two limitations: (i) they require collecting a large and varied training set to learn a low-dimensional set of basis functions, and (ii) they do not retain information about the 3D geometry of the object being imaged. In this paper, we show that it is possible to estimate low-dimensional manifolds that describe object appearance while retaining the geometrical information about the 3D structure of the object. By using a combination of analytically derived geometrical models and statistical learning methods, this can be achieved using a much smaller training set than most of the existing approaches. Specifically, we derive a quadrilinear manifold of object appearance that can represent the effects of illumination, pose, identity and deformation, and the basis functions of the tangent space to this manifold depend on the 3D surface normals of the objects. We show experimental results on constructing this manifold and how to efficiently track on it using an inverse compositional algorithm.

1. Introduction

Low dimensional representations of object appearance have proved to be one of the successful strategies in computer vision for applications in tracking, modeling and recognition. Active appearance models (AAMs) [4, 10], multilinear models [12, 13, 5, 14], and other low-dimensional manifold representations [9, 7] fall in this genre. These methods have two characteristics that may be limitations in many circumstances. *First*, in all these approaches, the construction of the underlying low-dimensional manifold relies upon obtaining different instances of the object's appearance under various conditions (e.g., pose, lighting, identity and deformations) and then using statistical data analysis tools to approximate the appearance space. This approach requires first collecting a large number of examples of the object's appearance and the accuracy of the method depends upon the examples that have been chosen for the training phase. Representation of appearances that have not been seen during the training phase can be inaccurate. *Second*, these representations do not retain any information about the 3D structure of the object, although the appearance must depend upon the 3D shape. This makes it difficult to understand the physical implication of the learned basis functions. In mathematical modeling terms, this is a purely *data-driven* approach.

In this paper, we show that it is possible to learn complex manifolds of object appearance that do not suffer from the above shortcomings. We term this as a "Geometry-Integrated Appearance Manifold" (GAM). The following are the main characteristics of the GAM.

- The GAM is a *quadrilinear* manifold of object appearance that is able to represent the combined effects of illumination, pose, identity and deformation.
- The basis vectors of the tangent space to this manifold depend upon the 3D surface normals of the object. Such a representation is not possible through methods that rely purely on learning-based approaches.
- The GAM is computed using a combination of analytically derived geometrical models and statistical learning methods. Thus, construction of the GAM requires significantly less data than AAM/ASM [4, 10], 3DMM [3], probabilistic appearance manifold (PAM) [9], and multilinear models (MLM) [13] (see Table 1 in Section 5). This also makes the learned manifold less dependent upon the actual examples that were used.
- The basis vectors of the GAM describe the local tangent space of this manifold, while many of the existing methods [13, 4, 10] derive a global subspace to represent the image appearance. Due to this locality property, the GAM has the potential to represent the image space more accurately.
- We show an application in using the inverse compositional (IC) algorithm to efficiently and accurately track objects on this manifold through changes of pose, lighting and deformations.
- Depending upon the application, it may be possible to de-

*The authors were partially supported by NSF grant IIS-0712253.

rive the manifold in a completely analytical manner, an example being tracking a rigid object (e.g., vehicle) through pose and lighting changes. In other examples, like face recognition, a combination of analytical approaches and statistical data analysis can be used for learning the manifold.

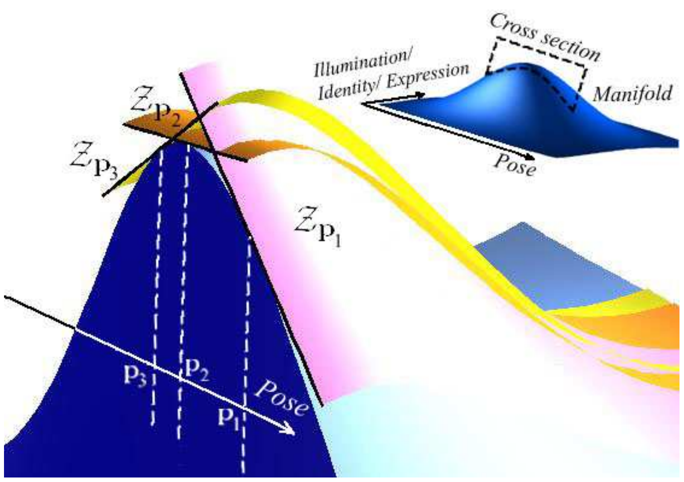


Figure 1. Pictorial representation of a GAM cross-section. Only two axes are shown for simplicity. The GAM can be visualized as a collection of locally linear tangent planes along the pose dimension.

2. Overview of Proposed Approach

There are two main parts of this paper - learning GAMs using a combination of geometrical models and statistical analysis, and adapting the IC algorithm for tracking and view synthesis using these manifolds.

2.1. Method for Learning GAMs

We combine analytically derived geometrical models that represent the effects of motion, lighting and 3D shape [2, 11, 16], with statistical learning approaches that are used to model the other effects like identity (e.g., faces of different people) and non-rigidity which are not easy to represent analytically. Lighting is modeled using a spherical harmonics based linear subspace representation [2, 11]. This is then combined with a recent result that proved the appearance of an image is bilinear in the 3D rigid motion and illumination parameters, with the 3D shape determining the basis vectors of the space [16]. The variations of this analytically derived bilinear basis over identity and deformation are then learned using multilinear SVD [8], and they together form a *quadrilinear* space of illumination, pose, identity and deformation. The GAM can be visualized (see Figure 1) as a collection of locally linear tangent planes along the pose dimension, where each tangent plane represents 3D motion in a local region around each pose. Thus the GAM is able to model the local tangent space around a pose.

The major difference of GAMs with other methods for computing appearance manifolds and subspaces [13, 5, 9, 15, 10, 7] is that the object appearance space is derived using a combination of geometrical models and data analysis tools, while the previous approaches relied purely on data analysis. This significantly reduces the data collection requirements for computing such manifolds, makes analysis on these manifolds less dependent on the actual data used to learn them in the first place, and allows representations of appearances that were not included in the learning phase. We will provide some concrete numerical examples to justify these in the experimental section. *Thus our method combines the precision and generalizability of model-based approaches with the robustness provided by statistical learning methods to deviations from the model predictions.*

2.2. Robust and Efficient Tracking on GAMs

We show how to track and synthesize novel views of an object using the learned GAMs. This is done by adapting the inverse compositional (IC) algorithm to the geometry of the manifold. We can estimate the 3D pose, lighting and deformation parameters of the object. The inverse compositional (IC) approach [1] is an efficient implementation of the Lucas-Kanade image alignment method and works by moving the expensive computation of gradients and Hessians out of an iterative loop. Due to 3D motion estimation in our case, the expensive computations of derivatives need to take place only at a few discrete poses (not once every frame).

Our tracking algorithm provides 3D estimates of motion, illumination model parameters, and identity and deformation parameters, thus going beyond illumination-invariant 2D tracking [6]. *It does not require a texture mapped 3D model of the object,* unlike many 3D model-based trackers. For tracking faces, we are able to obtain close to real-time performance while estimating 3D pose, lighting and expression parameters. Moreover, GAMs are easy to construct and can be built for non-face objects.

3. Method for Learning GAMs

We will start with the illumination representation of [2] and combine it with motion and shape using the results in [16] in order to derive an analytical representation of a low dimensional manifold of object appearance with variations in pose and lighting. We will then apply N-mode SVD, a multilinear generalization of SVD, to learn the variation of this manifold due to changes of identity and object deformations. We will show that the image appearance due to variations of illumination, pose, and deformation, is quadrilinear and compute the basis functions of this space.

3.1. Analytically Derived Manifold for Motion and Illumination - The Geometrical Approach

We start from the results in [16] which showed that it is possible to approximate the sequence of images of a moving object by a bilinear subspace of nine illumination coefficients and six motion variables [16]. Let the pose of the object in the camera reference frame to be defined as $\mathbf{p} = (\mathbf{T}^{\mathbf{T}}, \mathbf{\Omega}^{\mathbf{T}})^{\mathbf{T}}$. Representing by $\Delta\mathbf{T}$ the translation of the centroid of the object, by $\Delta\mathbf{\Omega}$ the rotation about the centroid, and by $\mathbf{l} \in \mathbb{R}^{N_l}$ ($N_l \approx 9$ for Lambertian objects with attached shadow) the illumination coefficients in a spherical harmonics basis (see [2] for details), the authors in [16] showed that under small motion, the reflectance image at $t_2 = t_1 + \delta t$ can be expressed as

$$I_{t_2}(\mathbf{u}) = \sum_{i=1}^{N_l} l_{i|t_2} b_{i|t_2}(\mathbf{u}), \tag{1}$$

where

$$b_{i|t_2}(\mathbf{u}) = b_{i|t_1}(\mathbf{u}) + \mathbf{A}_{t_1}(\mathbf{u},\mathbf{n})\Delta\mathbf{T}_{t_2} + \mathbf{B}_{t_1}(\mathbf{u},\mathbf{n})\Delta\mathbf{\Omega}_{t_2}. \tag{2}$$

In the above equations, $\mathbf{u}$ represents the image point projected from the 3D surface with surface normal $\mathbf{n}$, and $b_{i|t_1}(\mathbf{u})$ are the original basis images before motion. $\mathbf{A}_{t_1}$ and $\mathbf{B}_{t_1}$ contain the structure and camera intrinsic parameters, and are functions of $\mathbf{u}$ and the 3D surface normal $\mathbf{n}$. For each pixel $\mathbf{u}$, both $\mathbf{A}_{t_1}$ and $\mathbf{B}_{t_1}$ are N_l by 3 matrices.

It will be useful for us to represent this result using tensor notation as

$$\hat{\mathcal{I}}_{t_2} = \left(\mathcal{B}_{t_1} + \mathcal{C}_{t_1} \times_2 \begin{pmatrix} \Delta\mathbf{T}_{t_2} \\ \Delta\mathbf{\Omega}_{t_2} \end{pmatrix}\right) \times_1 \mathbf{l}_{t_2}, \tag{3}$$

where $\times_n$ is called the *mode-n product* [8] (Please refer to Appendix for detail of the *mode-n product*). For an image of size $M \times N$, $\mathcal{C}_{t_1}$ is a tensor of size $N_l \times 6 \times M \times N$. For each pixel (p,q) in the image, $\mathcal{C}_{klpq|t_1} = \begin{bmatrix} \mathbf{A}_{t_1}(\mathbf{u},\mathbf{n}) & \mathbf{B}_{t_1}(\mathbf{u},\mathbf{n}) \end{bmatrix}$ of size $N_l \times 6$, $\mathcal{B}_{t_1}$ is a sub-tensor of dimension $N_l \times 1 \times M \times N$, comprised of the basis images $b_{i|t_1}$, and $\mathcal{I}_{t_2}$ is a sub-tensor of dimension $1 \times 1 \times M \times N$, representing the image.[1]

3.2. Identity and Deformation Manifold - The Statistical Learning Approach

The above bilinear space of 3D motion and illumination is derived by using the knowledge of the 3D model of the object (tensor $\mathcal{C}$ contains the surface normals). However, the 3D shape is a function of the identity of the object (e.g., the identity of a face or a particular model of a car) and possible non-rigid deformations. The model in [16] cannot handle these cases. The challenge now is to generalize the above analytical model so that it can be used to represent a wide variety of appearances within a class of objects. We achieve this by learning multilinear appearance models.

Main Approach: Rather than directly modeling the variation in the appearance images, *we will model the bilinear bases of motion and illumination derived analytically in Section 3.1, and then combine all these different variations to obtain a multilinear model of object appearance. This will allow us to retain information about the geometry of the object.*

Using $[\bullet]_v$ to denote the vectorization operation, we can vectorize $\mathcal{B}$ and $\mathcal{C}$ in (3), and concatenate them, as

$$\mathbf{v} = \begin{bmatrix} [\mathcal{B}]_v \\ [\mathcal{C}]_v \end{bmatrix}. \tag{4}$$

Note that $\mathcal{B}$ and $\mathcal{C}$ can be obtained from the 3D model of the object. This $\mathbf{v}$ is the vectorized bilinear basis for one shape (i.e., one object) with dimension $I_v \times 1$, where $I_v = 7N_lMN$ (N_lMN for $\mathcal{B}$ and $6N_lMN$ for $\mathcal{C}$). Given the 3D shape of I_i objects with I_e different deformations, we can compute this vectorized bilinear basis $\mathbf{v}$ for every combination. For faces, these instances can be obtained from any 3D face modeling algorithm or by direct acquisition of 3D data. With the application to faces in mind, we will sometimes use the words deformation and expression interchangeably.

We use $\mathbf{v}_e^i$ to represent the vectorized bilinear basis of identity i with expression e. Let us rearrange them into a training data tensor $\mathcal{D}$ of size $I_i \times I_e \times I_v$ with the first dimension for identity, second dimension for expression (deformation) and the third dimension for the vectorized, analytically derived bilinear basis for each training sample. Applying the *N-Mode SVD* algorithm [8], the training data tensor can be decomposed as

$$\begin{aligned} \mathcal{D} &= \mathcal{Y} \times_1 U_i \times_2 U_e \times_3 U_v = \mathcal{Z} \times_1 U_i \times_2 U_e, \\ \text{where} \quad & \mathcal{Z} = \mathcal{Y} \times_3 U_v. \end{aligned} \tag{5}$$

$\mathcal{Y}$ is known as the core tensor of size $N_i \times N_e \times N_v$, and N_i and N_e are the number of bases we use for the identity and expression. With a slight abuse of terminology, we will call $\mathcal{Z}$, which is decomposed only along the identity and expression dimension with size $N_i \times N_e \times I_v$, to be the core tensor. U_i and U_e, with sizes of $I_i \times N_i$ and $I_e \times N_e$, are the left matrices of the SVD of

$$\mathcal{D}_{(1)} = \begin{pmatrix} \mathbf{v}_1^{1\,\mathbf{T}} & \dots & \mathbf{v}_{I_e}^{1\,\mathbf{T}} \\ & \dots & \\ \mathbf{v}_1^{I_i\,\mathbf{T}} & \dots & \mathbf{v}_{I_e}^{I_i\,\mathbf{T}} \end{pmatrix}$$

$$\text{and } \mathcal{D}_{(2)} = \begin{pmatrix} \mathbf{v}_1^{1\,\mathbf{T}} & \dots & \mathbf{v}_1^{I_i\,\mathbf{T}} \\ & \dots & \\ \mathbf{v}_{I_e}^{1\,\mathbf{T}} & \dots & \mathbf{v}_{I_e}^{I_i\,\mathbf{T}} \end{pmatrix}, \tag{6}$$

where the subscripts of tensor $\mathcal{D}$ indicate the tensor unfolding operation along the first and second dimension (please

[1]For the purposes of this paper, the general form of the expression suffice. Details can be found in [16].

refer to Appendix for detail of the tensor unfolding operation). According to the *N-mode SVD algorithm* and equation (5), the core tensor $\mathcal{Z}$ can be expressed as

$$\mathcal{Z} = \mathcal{D} \times_1 U_i^{\mathbf{T}} \times_2 U_e^{\mathbf{T}}. \tag{7}$$

3.3. Lighting, Motion, Identity and Deformation Manifold - Unifying Geometrical and Statistical Approaches

The core tensor $\mathcal{Z}$ contains the basis of identity and expression (or deformation) for $\mathbf{v}$ as

$$\mathbf{v}_i^{eT} = \mathcal{Z} \times_1 \mathbf{c}_i^{\mathbf{T}} \times_2 \mathbf{c}_e^{\mathbf{T}}, \tag{8}$$

where $\mathbf{c}_i$ and $\mathbf{c}_e$ are the coefficient vectors encoding the identity and expression. As $\mathbf{v}_i^e$ are the vectorized, bilinear basis functions of the illumination and 3D motion, the core tensor $\mathcal{Z}$ is *quadrilinear* in illumination, motion, identity and expression. As an example, this core tensor $\mathcal{Z}$ can describe all the face images of identity $\mathbf{c}_i$ with expression $\mathbf{c}_e$ and motion ($\Delta\mathbf{T}$, $\Delta\mathbf{\Omega}$) under illumination $\mathbf{l}$.

Due to the small motion assumption in the derivation of the analytical model of motion and illumination in Section 3.1, the core tensor $\mathcal{Z}$ can only represent the image of the object whose pose is close to the pose $\mathbf{p}$ under which the training samples of $\mathbf{v}$ are computed. To emphasize that $\mathcal{Z}$ is a function of pose $\mathbf{p}$, we denote it as $\mathcal{Z}_{\mathbf{p}}$ in the following derivation.

Since $\mathbf{v}$ is obtained by concatenating $[\mathcal{B}]_v$ and $[\mathcal{C}]_v$, $\mathcal{Z}_{\mathbf{p}}$ also contains two parts, $\mathcal{Z}_{\mathbf{p}}^{\mathcal{B}}$ with size $(N_i \times N_e \times N_l MN)$ and $\mathcal{Z}_{\mathbf{p}}^{\mathcal{C}}$ with size $(N_i \times N_e \times 6N_l MN)$. The first part encodes the variation of the image due to changes of identity, deformation and illumination at the pose $\mathbf{p}$, and the second part encodes the variation due to motion around $\mathbf{p}$, i.e., the tangent plane of the manifold along the motion direction. Rearranging the two sub-tensors according to the illumination and motion basis into sizes of $N_l \times 1 \times N_i \times N_e \times MN$ and $N_l \times 6 \times N_i \times N_e \times MN$ (this step is needed to undo the vectorization operation of equation (4)), we can represent the quadrilinear basis of illumination, 3D motion, identity, and deformation along the first, second, third and forth dimensions respectively.

The image with identity $\mathbf{c}_{i|t_2}$ and expression $\mathbf{c}_{e|t_2}$ after motion ($\Delta\mathbf{T}_{t_2}$,$\Delta\mathbf{\Omega}_{t_2}$) around pose $\mathbf{p}_{t_1}$ under illumination $\mathbf{l}_{t_2}$ can be obtained by

$$\begin{aligned}\mathcal{I}_{t_2} \quad = \quad & \mathcal{Z}_{\mathbf{p}_{t_1}}^{\mathcal{B}} \times_1 \mathbf{l}_{t_2} \times_3 \mathbf{c}_{i|t_2} \times_4 \mathbf{c}_{e|t_2} \\ & + \mathcal{Z}_{\mathbf{p}_{t_1}}^{\mathcal{C}} \times_1 \mathbf{l}_{t_2} \times_2 \begin{pmatrix} \Delta\mathbf{T}_{t_2} \\ \Delta\mathbf{\Omega}_{t_2} \end{pmatrix} \times_3 \mathbf{c}_{i|t_2} \times_4 \mathbf{c}_{e|t_2}. \end{aligned} \tag{9}$$

Note that we did not need examples of the object at different lighting conditions to construct this manifold. Also, the appearance variation due to rigid motion around each pose was modeled without any training examples. These parts of the manifold came from the analytical expressions in (3).

To represent the manifold at all the possible poses, we do not need such a tensor at every pose. Effects of 3D translation can be removed by centering and scale normalization, while in-plane rotation to a pre-defined pose can mitigate the effects of rotation about the z-axis. Thus, the image of object under arbitrary pose, $\mathbf{p}$, can always be described by the multilinear object representation at a pre-defined $(\mathbf{T}_x^{pd}, \mathbf{T}_y^{pd}, \mathbf{T}_z^{pd}, \mathbf{\Omega}_z^{pd})$, with only $\mathbf{\Omega}_x$ and $\mathbf{\Omega}_y$ depending upon the particular pose. Thus, the image manifold under any pose can be approximated by the collection of a few tangent planes on distinct $\mathbf{\Omega}_x^j$ and $\mathbf{\Omega}_y^j$, denoted as $\mathbf{p}_j$. In the following part, we will concentrate on the centered and scale normalized images.

4. Robust and Efficient Tracking on GAMs

An iterative gradient descent method for tracking using the GAM can be easily developed since we know the basis functions of the tangent space of the manifold. However, each iteration requires updating the bases, which is computationally expensive and requires knowledge of the 3D model of the object. The inverse compositional algorithm [1] works by moving these computational steps out of the iterative updating process. In addition, the constraint of knowing the 3D model of the object can be relaxed by reconstructing $\mathcal{B}$ and $\mathcal{C}$ from the core tensors $\mathcal{Z}^{\mathcal{B}}$ and $\mathcal{Z}^{\mathcal{C}}$. In keeping with standard notation used in tracking, we assume $\Delta t = 1$, and consider two frames at t and $t-1$.

From equation (9), the the cost function for estimation of 3D motion and lighting can be rewritten as

$$(\hat{\mathbf{l}}_t, \hat{\mathbf{m}}_t, \hat{\mathbf{c}}_{i|t}, \hat{\mathbf{c}}_{e|t}) = \arg \min_{\mathbf{l_t}, \mathbf{m_t}, \mathbf{c_{i|t}}, \mathbf{c_{e|t}}}$$

$$\left|\left| \mathcal{I}_t - \left(\mathcal{Z}_{\hat{\mathbf{p}}_{t-1}}^{\mathcal{B}} + \mathcal{Z}_{\hat{\mathbf{p}}_{t-1}}^{\mathcal{C}} \times_2 \mathbf{m}_t \right) \times_1 \mathbf{l}_t \times_3 \mathbf{c}_{i|t} \times_4 \mathbf{c}_{e|t} \right|\right|^2, \tag{10}$$

where $\mathbf{m}_t = \Delta\mathbf{p}_t = (\Delta\mathbf{T_t}^{\mathbf{T}}, \Delta\mathbf{\Omega}_t)^{\mathbf{T}}$. This cost function is quadrilinear in illumination, motion, identity and deformation variables. The optimization of (10) can be done by optimizing over each dimension one by one while keeping the others fixed. Starting from an initial pose estimate (where the manifold is approximated by a tangent), we will first optimize over illumination, identity and expression dimensions, and then apply the inverse compositional algorithm for optimization over motion.

4.1. Warping Function for Inverse Compositional (IC) Estimation on GAMs

Consider an input frame $I_t(\mathbf{u})$ at time instance t with image coordinate $\mathbf{u}$. We introduce a warp operator $\mathbf{W_p} : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ such that, if the pose of $I_t(\mathbf{u})$ is $\mathbf{p}_t$, the pose of $I_t(\mathbf{W_{p_t}}(\mathbf{u}, \mathbf{m_t}))$ is $\mathbf{p}_t + \mathbf{m}_t$. Basically, $\mathbf{W_p}$ represents the displacement in the image plane due to a pose transformation of the 3D model (see Fig. 2). Denote the pose transformed image $I_t(\mathbf{W_{p_t}}(\mathbf{u}, \mathbf{m_t}))$ in tensor notation $\tilde{\mathcal{I}}_t^{\mathbf{W_{p_t}}(\mathbf{m_t})}$.

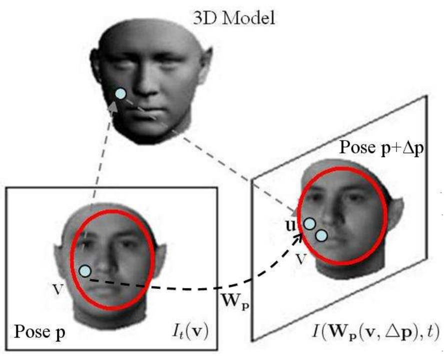


Figure 2. Illustration of the warping function $\mathbf{W}$. A point $\mathbf{v}$ in image plane is projected onto the surface of the 3D object model. After the pose transformation with $\triangle\mathbf{p}$, the point on the surface is back projected onto the image plane at a new point $\mathbf{u}$. The warping function maps from $\mathbf{v} \in \mathbb{R}^2$ to $\mathbf{u} \in \mathbb{R}^2$. The red ellipses show the common part in both frames that the warping function $\mathbf{W}$ is defined upon.

Using this warp operator and ignoring the regularization term, we can restate the cost function (10) in the inverse compositional framework as

$$(\hat{\mathbf{l}}_t, \hat{\mathbf{m}}_t, \hat{\mathbf{c}}_{i|t}, \hat{\mathbf{c}}_{e|t}) = \arg \min_{\mathbf{l},\mathbf{m},\mathbf{c_i},\mathbf{c_e}}$$
$$\|\tilde{\mathcal{I}}_t^{\mathbf{W}_{\hat{\mathbf{p}}_{\mathbf{t-1}}}(-\hat{\mathbf{m}}_\mathbf{t})} - \mathcal{Z}^{\mathcal{B}}_{\hat{\mathbf{p}}_{t-1}} \times_1 \mathbf{l}_t \times_3 \mathbf{c}_{i|t} \times_4 \mathbf{c}_{e|t}\|^2. \quad (11)$$

Given the other parameters of the quadrilinear manifold, the cost function can be minimized over $\mathbf{m}_t$ by iteratively solving for increments $\triangle\mathbf{m}$ in

$$\|\tilde{\mathcal{I}}_t^{\mathbf{W}_{\hat{\mathbf{p}}_{\mathbf{t-1}}}(-\mathbf{m}_\mathbf{t})} - \left(\mathcal{Z}^{\mathcal{B}}_{\hat{\mathbf{p}}_{t-1}} + \mathcal{Z}^{\mathcal{C}}_{\hat{\mathbf{p}}_{t-1}} \times_2 \triangle\mathbf{m}\right) \times_1 \mathbf{l}_t \times_3 \mathbf{c}_{i|t} \times_4 \mathbf{c}_{e|t}\|^2. (12)$$

In each iteration, $\mathbf{m}_t$ is updated such that $\mathbf{W}_{\hat{\mathbf{p}}_{t-1}}(\mathbf{u}, -\mathbf{m_t}) \leftarrow \mathbf{W}_{\hat{\mathbf{p}}_{t-1}}(\mathbf{u}, -\mathbf{m_t}) \circ \mathbf{W}_{\hat{\mathbf{p}}_{t-1}}(\mathbf{u}, \triangle\mathbf{m})^{-1}$ (please refer to the Appendix for the definition of the compositional operator $\circ$ and the inverse of the warp, $\mathbf{W}^{-1}$). Using the additivity of pose transformation for small $\triangle\mathbf{m}$, $\mathbf{W}_{\hat{\mathbf{p}}_{\mathbf{t-1}}}(\mathbf{W}_{\hat{\mathbf{p}}_{\mathbf{t-1}}}(\mathbf{u}, \triangle\mathbf{m})^{-1}, -\mathbf{m_t}) = \mathbf{W}_{\hat{\mathbf{p}}_{\mathbf{t-1}}}(\mathbf{W}_{\hat{\mathbf{p}}_{\mathbf{t-1}}+\triangle\mathbf{m}}(\mathbf{u}, -\triangle\mathbf{m}), -\mathbf{m_t}) = \mathbf{W}_{\hat{\mathbf{p}}_{\mathbf{t-1}}+\triangle\mathbf{m}}(\mathbf{u}, -\triangle\mathbf{m} - \mathbf{m_t}) \approx \mathbf{W}_{\hat{\mathbf{p}}_{\mathbf{t-1}}}(\mathbf{u}, -\triangle\mathbf{m} - \mathbf{m_t})$. Thus, the above update is essentially $\mathbf{m}_t \leftarrow \mathbf{m}_t + \triangle\mathbf{m}$.

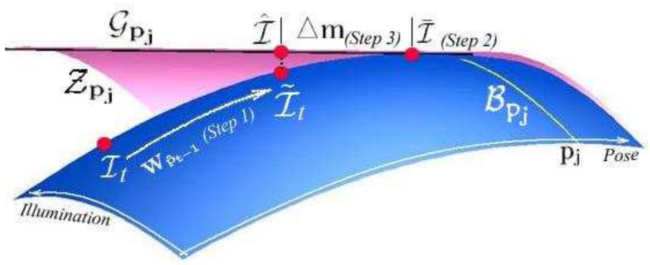

$\mathcal{G}_{\mathbf{p_j}}$
$\hat{\mathcal{I}}$
$\triangle\mathbf{m}$ (Step 3)
$\bar{\mathcal{I}}$ (Step 2)
$\mathcal{Z}_{\mathbf{p_j}}$
$\tilde{\mathcal{I}}_t$
$\mathcal{B}_{\mathbf{p_j}}$
$\mathcal{I}_t$
$\mathbf{W}_{\hat{\mathbf{p}}_{t-1}}$ (Step 1)
$\mathbf{p_j}$
Pose
Illumination

Figure 3. Pictorial representation of the inverse compositional tracking scheme on GAMs.

In [1], the authors proved that, for the inverse compositional algorithm to be provably equivalent to the Lucas-Kanade algorithm to the first order approximation of $\triangle\mathbf{m}$, the set of warps $\{\mathbf{W}\}$ must form a group, i.e. every warp $\mathbf{W}$ must be invertible. If the change of pose is small enough, the visibility for most of the pixels will remain the same - thus $\mathbf{W}_{\hat{\mathbf{p}}_{t-1}}$ can be considered approximately invertible. However, if the pose change becomes too big, some portion of the object will become invisible after the pose transformation, and $\mathbf{W}_{\hat{\mathbf{p}}_{t-1}}$ will no longer be invertible. A rigorous proof of convergence is available in [17].

Since the GAM along the motion direction is composed of a set of tangent planes at a few discrete poses (see Figure 1), the computations for $\triangle\mathbf{m}$ need to happen only at these poses (called cardinal poses). Thus all frames that are close to a particular pose $\mathbf{p}_j$ will use the $\mathcal{B}$ and $\mathcal{C}$ at that pose, and the warp $\mathbf{W}_{\hat{\mathbf{p}}_{t-1}}$ should be performed to normalize the pose to $\mathbf{p}_j$. While most of the existing inverse compositional methods move the expensive update steps out of the iterations for two-frame matching, we go even further and perform these expensive computations only once every few frames. This is because we estimate 3D motion.

4.2. The IC Algorithm on GAMs

Due to the fact that GAM is constructed at the pre-defined values of $(\mathbf{T}^{pd}_x, \mathbf{T}^{pd}_y, \mathbf{T}^{pd}_z, \mathbf{\Omega}^{pd}_z)$, for each input frame I_t, we will use the pose estimated at t-1, i.e. $\hat{\mathbf{p}}_{t-1}$, for normalizing the image to these pre-defined values. Then, the Inverse Compositional algorithm with GAM will be applied on these normalized images. The 3D estimates will be combined with the above values to obtain the final pose estimates.

A pictorial representation of the IC tracking algorithm on GAMs is shown in Fig. 3. Consider a sequence of image frames $\mathcal{I}_t, t = 0, ..., N-1$. Assume that we know the pose and illumination estimates for frame $t-1$, i.e., $\hat{\mathbf{p}}_{t-1}$ and $\hat{\mathbf{l}}_{t-1}$.

• *Step 1*. For the new input frame $\mathcal{I}_t$, normalize it to the pre-defined values $(\mathbf{T}^{pd}_x, \mathbf{T}^{pd}_y, \mathbf{T}^{pd}_z, \mathbf{\Omega}^{pd}_z)$ using the pose estimates at $t-1$, i.e. $\hat{\mathbf{p}}_{t-1}$. Find the closest $\mathbf{p_j}$ to $\hat{\mathbf{p}}^{pd}_{t-1} \triangleq (\mathbf{T}^{pd}_x, \mathbf{T}^{pd}_y, \mathbf{T}^{pd}_z, \hat{\mathbf{\Omega}}_{x|t-1}, \hat{\mathbf{\Omega}}_{y|t-1}, \mathbf{\Omega}^{pd}_z)$. Assume motion $\hat{\mathbf{m}}^{pd}_t$ at this pre-defined pose to be zero, illumination condition $\hat{\mathbf{l}}_t = \hat{\mathbf{l}}_{t-1}$, identity coefficient $\hat{\mathbf{c}}_{i|t} = \hat{\mathbf{c}}_{i|t-1}$, and expression coefficient $\hat{\mathbf{c}}_{e|t} = \hat{\mathbf{c}}_{e|t-1}$.

• *Step 2*. Apply the pose transformation operator $\mathbf{W}_{\hat{\mathbf{p}}^{pd}_{t-1}}$ to get the pose normalized version of the frame $\tilde{\mathcal{I}}^{\mathbf{W}_{\hat{\mathbf{p}}^{pd}_{t-1}}(\mathbf{p_j} - \hat{\mathbf{p}}^{\mathbf{pd}}_{\mathbf{t-1}} - \hat{\mathbf{m}}^{\mathbf{pd}}_{\mathbf{t}})}$, i.e., $I_t(\mathbf{W}_{\hat{\mathbf{p}}^{pd}_{t-1}}(\mathbf{u}, \mathbf{p_j} - \hat{\mathbf{p}}^{pd}_{t-1} - \hat{\mathbf{m}}^{pd}_t))$. This is shown in Figure 3, where the input frame $\mathcal{I}_t$ on the manifold is first normalized and warped to $\tilde{\mathcal{I}}_t$ which is within a nearby region of pose $\mathbf{p_j}$.

• *Step 3*. Use (15) to alternately estimate $\hat{\mathbf{l}}_t$, $\hat{\mathbf{c}}_{i|t}$ and $\hat{\mathbf{c}}_{e|t}$

of the pose normalized image $\mathcal{I}_t^{\mathbf{W}_{\hat{\mathbf{p}}_{t-1}^{pd}}(\mathbf{u},\mathbf{p_j}-\hat{\mathbf{p}}_{t-1}^{pd}-\hat{\mathbf{m}}_t^{pd})}$ as follows.
Using (8), $\mathcal{B}_{\mathbf{p_j}}$ can be written as

$$\mathcal{B}_{\mathbf{p_j}} = \left[\mathcal{Z}_{\mathbf{p_j}}^{\mathcal{B}} \times_3 \hat{\mathbf{c}}_{i|t} \times_4 \hat{\mathbf{c}}_{e|t}\right]_v^{-1}. \tag{13}$$

Denoting the basis for the identity and expression as $\mathcal{E}$ and $\mathcal{F}$, we can similarly compute them as

$$\mathcal{E}_{\mathbf{p_j}} = \left[\mathcal{Z}_{\mathbf{p_j}}^{\mathcal{B}} \times_1 \hat{\mathbf{l}}_t \times \hat{\mathbf{c}}_{e|t}\right]_v^{-1}, \mathcal{F}_{\mathbf{p_j}} = \left[\mathcal{Z}_{\mathbf{p_j}}^{\mathcal{C}} \times_1 \hat{\mathbf{l}}_t \times_3 \hat{\mathbf{c}}_{i|t}\right]_v^{-1}. \tag{14}$$

Thus the illumination coefficients can be estimated using least squares (since the illumination bases after motion (2) are not orthogonal), while the identity and expression coefficients can be estimated by projection of the image onto the corresponding basis as

$$\hat{\mathbf{l}}_t = (\mathcal{B}_{\mathbf{p}_j}\mathcal{B}_{\mathbf{p}_j}^{\mathbf{T}})^{-1}\mathcal{B}_{\mathbf{p}_j}^{\mathbf{T}}\mathcal{I}_{t(1)}, \hat{\mathbf{c}}_{i|t} = \mathcal{E}_{\mathbf{p}_j}^{\mathbf{T}}\mathcal{I}_{t(1)}, \hat{\mathbf{c}}_{e|t} = \mathcal{F}_{\mathbf{p}_j}^{\mathbf{T}}\mathcal{I}_{t(1)}. \tag{15}$$

Iteratively solving for $\hat{\mathbf{l}}$, $\hat{\mathbf{c}}_i$ and $\hat{\mathbf{c}}_e$, the cost function (11) is minimized over illumination, identity and expression directions. In Figure 3, the curve $\mathcal{B}_{\mathbf{p_j}}$ shows the manifold of the image at pose $\mathbf{p_j}$ with motion as zero, but varying illumination, identity or deformation. By iteratively minimizing along the illumination, identity, and deformation directions, we find the point

$$\bar{\mathcal{I}}_t = \mathcal{Z}_{\mathbf{p}_j}^{\mathcal{B}} \times_1 \hat{\mathbf{l}}_t \times_3 \hat{\mathbf{c}}_{i|t} \times_4 \hat{\mathbf{c}}_{e|t} \tag{16}$$

on the curve $\mathcal{B}_{\mathbf{p_j}}$ which has the minimum distance to the pose normalized point $\tilde{\mathcal{I}}_t$.

• *Step 4*. With the estimated $\hat{\mathbf{l}}_t$, $\hat{\mathbf{c}}_{i|t}$ and $\hat{\mathbf{c}}_{e|t}$ from Step 3, use (18) to estimate the motion increment $\triangle\mathbf{m}^{pd}$. Update $\hat{\mathbf{m}}_t^{pd}$ with $\hat{\mathbf{m}}_t^{pd} \leftarrow \hat{\mathbf{m}}_t^{pd} + \triangle\mathbf{m}^{pd}$. This can be done as follows. Rewrite the cost function in (12) at the cardinal pose $\mathbf{p}_j$ as

$$\left\|\tilde{\mathcal{I}}_t^{\mathbf{W}_{\hat{\mathbf{p}}_{t-1}^{pd}}(\mathbf{p}_j-\hat{\mathbf{p}}_{t-1}^{pd}-\hat{\mathbf{m}}_t^{pd})} - \left(\bar{\mathcal{I}}_t + \mathcal{G}_{\mathbf{p_j}}^{\mathbf{T}}\triangle\mathbf{m}^{pd}\right)\right\|^2,$$
$$\text{where } \mathcal{G}_{\mathbf{p_j}} \triangleq \left[\mathcal{Z}_{\mathbf{p_j}}^{\mathcal{C}} \times_1 \hat{\mathbf{l}}_t \times \hat{\mathbf{c}}_{i|t} \times \hat{\mathbf{c}}_{e|t}\right]_v^{-1}. \tag{17}$$

$\mathcal{G}_{\mathbf{p_j}}$ is the motion basis at pose $\mathbf{p_j}$ with fixed $\hat{\mathbf{l}}_t$, $\hat{\mathbf{c}}_{i|t}$ and $\hat{\mathbf{c}}_{e|t}$. Recall that $\mathcal{Z}_{\mathbf{p_j}}^{\mathcal{C}}$ is a tensor of size $N_l \times 6 \times N_i \times N_e \times MN$ - thus $\mathcal{G}_{\mathbf{p_j}}$ degenerates to a matrix of size $6 \times MN$. In Figure 3, we compute the tangent along the motion direction, shown as the black line $\mathcal{G}_{\mathbf{p_j}}$, from the core tensor shown as the surface $\mathcal{Z}$.

Taking the derivative of (17) with respect to $\triangle\mathbf{m}^{pd}$, and setting it to be zero, we have

$$\triangle\mathbf{m}^{pd} = \left[\mathcal{G}_{\mathbf{p_j}}\mathcal{G}_{\mathbf{p_j}}^{\mathbf{T}}\right]^{-1}\mathcal{G}_{\mathbf{p_j}}(\tilde{\mathcal{I}}_t^{\mathbf{W}_{\hat{\mathbf{p}}_{t-1}^{pd}}(\mathbf{p}_j-\hat{\mathbf{p}}_{t-1}^{pd}-\hat{\mathbf{m}}_t^{pd})} - \bar{\mathcal{I}}_t), \tag{18}$$

and the motion estimates $\hat{\mathbf{m}}_t^{pd}$ should be updated with the increments $\hat{\mathbf{m}}_t^{pd} \leftarrow \hat{\mathbf{m}}_t^{pd} + \triangle\mathbf{m}^{pd}$. The overall computational cost is reduced significantly by making the gradient $\mathcal{G}_{\mathbf{p_j}}$ independent of the updating variable $\hat{\mathbf{m}}_t^{pd}$. In Figure 3, $\triangle\mathbf{m}^{pd}$ is shown to be the distance from point $\bar{\mathcal{I}}_t$ to $\hat{\mathcal{I}}_t$, the projection of $\tilde{\mathcal{I}}_t$, onto the motion tangent.

• *Step 5*. Use the updated $\hat{\mathbf{m}}_t^{pd}$ from Step 4 to update the pose normalized image as $\tilde{\mathcal{I}}_t^{\mathbf{W}_{\hat{\mathbf{p}}_{\mathbf{t-1}}^{\mathbf{pd}}}(\mathbf{p_j}-\hat{\mathbf{p}}_{\mathbf{t-1}}-\hat{\mathbf{m}}_{\mathbf{t}}^{\mathbf{pd}})}$, i.e. $I_t(\mathbf{W}_{\hat{\mathbf{p}}_{t-1}^{pd}}(\mathbf{u}, \mathbf{p_j} - \hat{\mathbf{p}}_{t-1} - \hat{\mathbf{m}}_t^{pd}))$.

• *Step 6*. Repeat Steps 2, 3, 4 and 5 for that input frame till the difference error ε between the pose normalized image $\tilde{\mathcal{I}}_t^{\mathbf{W}_{\hat{\mathbf{p}}_{t-1}^{pd}}(\mathbf{p_j}-\hat{\mathbf{p}}_{t-1}^{pd}-\hat{\mathbf{m}}_t^{pd})}$ and the rendered image $\bar{\mathcal{I}}_t$ can be reduced below an acceptable threshold.

• Step 7. Undo the normalization of Step 1 to inverse transform $\hat{\mathbf{m}}_t^{pd}$ to $\hat{\mathbf{m}}_t$ and update $\hat{\mathbf{p}}_t = \hat{\mathbf{p}}_{t-1} + \hat{\mathbf{m}}_t$.

• Step 8. Set t = t + 1. Repeat Steps 1, 2, 3, 4, 5, 6 and 7. Continue till t = N - 1.

5. Experiment Results

A. Analysis of the GAM: As discussed above, the advantages of using the GAMs are (i) ease of construction due to the need for significantly less number of training images, (ii) ability to represent objects at all poses and lighting conditions from only a few examples during training, and (iii) accuracy and efficiency of tracking. We will now show results to justify these claims.

• **Constructing GAM of faces:** In the case of faces, we will need at least one image for every person. We then estimate the face model and compute the vectorized tensor $\mathbf{v}$ at a pre-defined collection of poses $\mathbf{p}_j$. For each expression, we will need at least one image per person. Thus for N_i people with N_e expressions, we need N_iN_e images. In our experiments, $N_i = 100$ and $N_e = 7$ thus requiring 700 images for all the people and every expression. To compare with other methods modeling the appearance manifolds, we list the number of the exemple images needed for training in Table 1. Moreover, *the GAM can model the appearance space not only at these discrete poses, but also the manifold in a local region around each pose.* In our experiments, the pose collection $\mathbf{p}_j$ is chosen to be every 15° along the vertical rotational axis, and every 20° along the horizontal rotational axis.

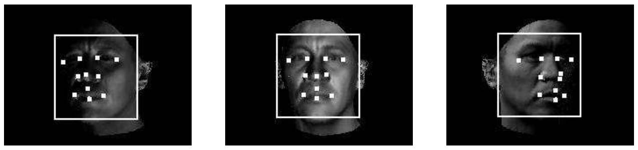
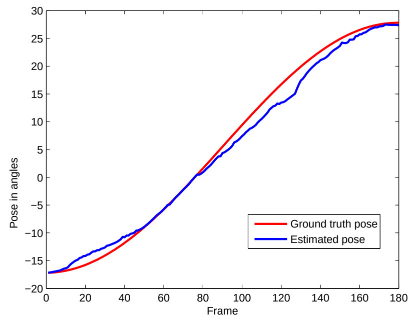
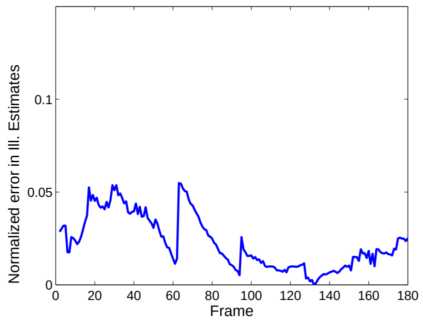


Figure 4. The back projection of the feature points on the generated 3D face model using the estimated 3D motion onto some input frames.

• **Accuracy of motion and illumination estimates:** We will now show some results on the accuracy of tracking on the GAM with known ground truth. We use the 3DMM



(a)



(b)

Figure 5. (a): 3D estimates (blue) and ground truth (red) of pose against frames. (b): The normalized error of the illumination estimates vs. frame numbers.

[3] to generate a face. The generated face model is rotated along the vertical axis at some specific angular velocity, and the illumination is changed both in direction (from right-bottom corner to the left-top corner) and in brightness (from dark to bright to dark). In Figure 4, the images show the back projection of some feature points on the 3D model onto the input frames using the estimated motion under three different illumination conditions. In Figure 5, (a) shows the comparison between the estimated motion (in blue) and the ground truth (in red). The maximum error in pose estimates is $3.57°$ and the average error is $1.22°$. Figure 5 (b) shows the norm of the error between the ground truth illumination coefficients and the estimated ones from the GAM, normalized with the ground truth (we cannot show all the illumination coefficients due to lack of space. The maximum error is 5.5% and the average is 2.2%. The peaks in the error plot are due to the change of the cardinal pose $\mathbf{p}_j$ (the tangent planes along the pose dimension).

Table 1. Comparison of the size of the training set needed for constructing face appearance models

AAM [4]	One image per person, but illumination and expression are not modeled. Pose variation is achieved through a shape normalization warping.
PAM [9]	300 images per person modeling only pose variation.
Multilinear model [13]	225 images per person modeling pose and illumination variations. No expression is modeled.
GAM	One image per person while modeling both pose and illumination variation. When modeling N_e kinds of expressions, N_e images per person needed.

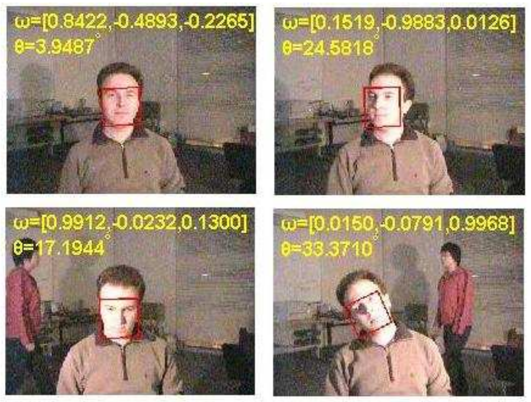

ω=[0.8422,-0.4893,-0.2265]
θ=3.9487°
ω=[0.1519,-0.9883,0.0126]
θ=24.5818°
ω=[0.9912,-0.0232,0.1300]
θ=17.1944°
ω=[0.0150,-0.0791,0.9968]
θ=33.3710°

Figure 6. An example of face tracking using GAMs under changes of pose and lighting. The estimated pose is shown on the top of the frames. (Should be viewed on a monitor)

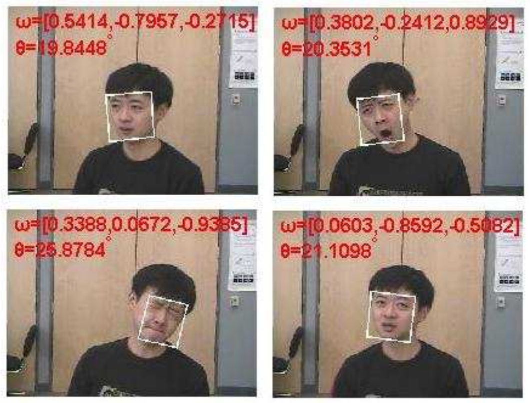

ω=[0.5414,-0.7957,-0.2715]
θ=19.8448°
ω=[0.3802,-0.2412,0.8929]
θ=20.3531°
ω=[0.3388,0.0672,-0.9385]
θ=25.8784°
ω=[0.0603,-0.8592,-0.5082]
θ=21.1098°

Figure 7. Another example of face tracking using GAMs under changes of pose and expressions. The estimated pose is shown on the top of the frames.

B. IC Tracking on GAM using Real Data: Figure 6 and Figure 7 show results of face tracking under large changes of pose, lighting, expression and background using the IC approach. The images in Figure 6 show tracking under illumination variations. The images in Figure 7 show tracking on the GAM with expression variations. On the top of the frames, we show the estimated pose of the face at the current frame. The pose is represented as a unit vector for the rotation axis, and the rotation angle in degrees, where the reference is taken to be the frontal face (i.e., we can get the rotation matrix $\mathbf{R} = e^{\hat{\omega}\theta}$). We did not require a texture-mapped 3D model as is common in many 3D model-based tracking methods. Our method outputs not only the 2D locations of the face (which is shown in the figures) but also the 3D pose (shown in figure), expression, and lighting parameters. We are able to obtain close to real-time performance using a MATLAB implementation.

6. Conclusions

In this paper, we showed that it is possible to estimate low-dimensional manifolds that describe object appearance with a small number of training samples using a combination of analytically derived geometrical models and statistical data analysis. We derived a quadrilinear space of object appearance that can represent the effects of illumination, motion, identity and deformation, and termed it as the Geometry-Integrated Appearance Manifold. We showed specific examples on how to construct this manifold and demonstrated the accuracy of pose and lighting estimation.

Appendix

Mode-N Product: The *mode-n product* of a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_n \times \ldots \times I_N}$ by a vector $\mathbf{V} \in \mathbb{R}^{1 \times I_n}$, denoted by $\mathcal{A} \times_n \mathbf{V}$, is the $I_1 \times I_2 \times \ldots \times 1 \times \ldots \times I_N$ tensor

$$(\mathcal{A} \times_n \mathbf{V})_{i_1 \ldots i_{n-1} 1 i_{n+1} \ldots i_N} = \sum_{i_n} a_{i_1 \ldots i_{n-1} i_n i_{n+1} \ldots i_N} v_{i_n}.$$

Tensor Unfolding Operation: Assume an Nth-order tensor $\mathcal{A} \in \mathbf{C}^{I_1 \times I_2 \times \ldots \times I_N}$. The matrix unfolding $\mathbf{A}_{(n)} \in \mathbf{C}^{I_n \times (I_{n+1} I_{n+2} \ldots I_N I_1 I_2 \ldots I_{n-1})}$ contains the element $a_{i_1 i_2 \ldots i_N}$ at the position with row number i_n and column number equal to $(i_{n+1} - 1) I_{n+2} I_{n+3} \ldots I_N I_1 I_2 \ldots I_{n-1} + (i_{n+2} - 1) I_{n+3} I_{n+4} \ldots I_N I_1 I_2 \ldots I_{n-1} + \cdots + (i_N - 1) I_1 I_2 \ldots I_{n-1} + (i_1 - 1) I_2 I_3 \ldots I_{n-1} + \cdots + i_{n-1}$.

Compositional Operator: The compositional operator $\circ$ means the second warp is composed into the first warp, i.e. $\mathbf{W}_{\hat{\mathbf{p}}_{t-1}}(\mathbf{u}, -\mathbf{m}) \equiv \mathbf{W}_{\hat{\mathbf{p}}_{t-1}}(\mathbf{W}_{\hat{\mathbf{p}}_{t-1}}(\mathbf{u}, \triangle\mathbf{m})^{-1}, -\mathbf{m})$.

The Inverse of Warp: The inverse of the warp $\mathbf{W}$ is defined to be the $\mathbb{R}^2 \rightarrow \mathbb{R}^2$ mapping such that if we denote the pose of $I_t(\mathbf{v})$ as $\mathbf{p}$, the pose of $I_t(\mathbf{W}_\mathbf{p}(\mathbf{W}_\mathbf{p}(\mathbf{v}, \triangle\mathbf{p}), \triangle\mathbf{p})^{-1})$ is $\mathbf{p}$ itself. As the warp $\mathbf{W}_\mathbf{p}(\mathbf{v}, \triangle\mathbf{p})$ transforms the pose from $\mathbf{p}$ to $\mathbf{p} + \triangle\mathbf{p}$, the inverse $\mathbf{W}_\mathbf{p}(\mathbf{v}, \triangle\mathbf{p})^{-1}$ should transform the pose from $\mathbf{p} + \triangle\mathbf{p}$ to $\mathbf{p}$, i.e. $\mathbf{W}_\mathbf{p}(\mathbf{v}, \triangle\mathbf{p})^{-1} = \mathbf{W}_{\mathbf{p}+\triangle\mathbf{p}}(\mathbf{v}, -\triangle\mathbf{p})$. Thus $\{\mathbf{W}_\mathbf{p}\}$ is a group.

References

[1] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255, February 2004. 2, 4, 5

[2] R. Basri and D. Jacobs. Lambertian Reflectance and Linear Subspaces. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(2):218–233, February 2003. 2, 3

[3] V. Blanz and T. Vetter. Face recognition based on fitting a 3D morphable model. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(9):1063–1074, Sep. 2003. 1, 7

[4] T. Cootes, G. Edwards, and C. Taylor. Active Appearance Models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(6):681–685, June 2001. 1, 7

[5] A. Elgammal and C. Lee. Separating style and content on a nonlinear manifold. In *Computer Vision and Pattern Recognition*, pages I: 478–485, 2004. 1, 2

[6] G. D. Hager and P. Belhumeur. Efficient Region Tracking With Parametric Models of Geometry and Illumination. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(10):1025–1039, 1998. 2

[7] G. W. J.-M. F. Hua Yang, Marc Pollefeys and A. Ilie. Differential Camera Tracking through Linearizing the Local Appearance Manifold. *IEEE International Conference on Computer Vision and Pattern Recognition*, 2007. 1, 2

[8] L. D. Lathauwer, B. D. Moor, and J. Vandewalle. A Multillinear Singular Value Decomposition. *SIAM J. Matrix Anal. Appl.*, 21(4):1253–1278, 2000. 2, 3

[9] K. Lee, J. Ho, M. Yang, and D. Kriegman. Video-based face recognition using probabilistic appearance manifolds. In *Computer Vision and Pattern Recognition*, pages I: 313–320, 2003. 1, 2, 7

[10] I. Matthews and S. Baker. Active appearance models revisited. *International Journal of Computer Vision*, 60(2):135 – 164, November 2004. 1, 2

[11] R. Ramamoorthi and P. Hanrahan. A signal-processing framework for inverse rendering. In E. Fiume, editor, *SIGGRAPH 2001, Computer Graphics Proceedings*, pages 117–128. ACM Press / ACM SIGGRAPH, 2001. 2

[12] J. B. Tenenbaum and W. T. Freeman. Separating style and content with bilinear models. *Neural Computation*, 12(6):1247–1283, 2000. 1

[13] M. Vasilescu and D. Terzopoulos. Multilinear Independent Components Analysis. In *Computer Vision and Pattern Recognition*, 2005. 1, 2, 7

[14] D. Vlasic, M. Brand, H. Pfister, and J. Popovic. Face transfer with multilinear models. *ACM Transactions on Graphics(TOG)*, pages 426–433, 2005. 1

[15] H. Wang and N. Ahuja. Facial expression decomposition. *IEEE International Conference on Computer Vision*, 2:958 – 965, 2003. 2

[16] Y. Xu and A. Roy-Chowdhury. Integrating Motion, Illumination and Structure in Video Sequences, With Applications in Illumination-Invariant Tracking. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, In Press. 2, 3

[17] Y. Xu and A. Roy-Chowdhury. Inverse compositional estimation of 3d motion and lighting in dynamic scenes. *IEEE Trans. on Pattern Analysis and Machine Intellitence*, In Press. 5