Aligning Sequences and Actions by Maximizing Space-Time Correlations

Yaron Ukrainitz and Michal Irani

Department of Computer Science and Applied Mathematics
The Weizmann Institute of Science
Rehovot, Israel

Abstract. We introduced an algorithm for sequence alignment, based on maximizing local space-time correlations. Our algorithm aligns sequences of the same action performed at different times and places by different people, possibly at different speeds, and wearing different clothes. Moreover, the algorithm offers a unified approach to the problem of sequence alignment for a wide range of scenarios (e.g., sequence pairs taken with stationary or jointly moving cameras, with the same or different photometric properties, with or without moving objects). Our algorithm is applied directly to the dense space-time intensity information of the two sequences (or to filtered versions of them). This is done without prior segmentation of foreground moving objects, and without prior detection of corresponding features across the sequences. Examples of challenging sequences with complex actions are shown, including ballet dancing, actions in the presence of other complex scene dynamics (clutter), as well as multi-sensor sequence pairs.

1 Introduction

Given two video sequences of a dynamic scene, the problem of sequence alignment is defined as finding the spatial and temporal coordinate transformation that brings one sequence into alignment with the other, both in space and in time. In this work we focus on the alignment of sequences with similar dynamics, but with significantly different appearance properties. In particular, we address two applications in a single unified framework:
1. *Action Alignment*: The same action is performed at different times and places by different people, possibly at different speeds, and wearing different clothes (optionally with different sensors). We would like to recover the space-time transformation which best aligns the actions (the foreground moving object), regardless of their backgrounds or other dynamic scene clutter.
2. *Multi-Sensor Alignment*: The same dynamic scene is recorded *simultaneously* by multiple cameras (of same or of different sensing modalities). In this case (of simultaneous recording) we would like to bring into alignment the entire scene (both the foreground moving objects and the background scene).

While sequences obtained by different sensors have significantly different spatial appearances, their temporal properties (scene or camera motion, trajectories of moving objects, etc.) are usually invariant to the sensing modalities, and are therefore shared by the two sequences. The same observation is true also for sequences of the same action performed by different people at different times and places. Such temporal changes are not captured in any individual frame. They are, however, contained in the space-time volumes generated by the two sequences. Sequence-to-sequence alignment is therefore a more powerful approach to handle those difficult scenarios than image-to-image alignment.

Several approaches to sequence alignment were suggested. Most of these methods assume that the video sequences are recorded simultaneously. Moreover, they are restricted to a particular scenario (e.g., moving objects [5], moving cameras [3], similar appearance properties [4]). Moreover, none of these methods is applicable to alignment of actions performed at different times and places.

Methods for aligning actions were also suggested (e.g., [10, 6, 2]). However, these require manual selection of corresponding feature points across the sequences. Some of them provide only temporal synchronization. In [11] an approach was proposed for detecting behavioral correlations in video under spatial and temporal *shifts*. Its output is a coarse space-time correlation volume. This approach does not account for spatial nor temporal scaling (nor more complex geometric deformations), nor was it used for aligning video clips (since video alignment requires sub-pixel and sub-frame accuracy).

In this paper we propose a unified approach to sequence alignment which is suited both for sequences recorded simultaneously (for a variety of scenarios), as well as for action sequences. Our approach is inspired by the multi-sensor image-alignment method presented in [8]. We extend it into space-time, and take it beyond multi-sensor alignment, to alignment of actions. Alignment in space and time is obtained by maximizing the local space-time correlations between the two sequences. Our method is applied directly to the dense space-time intensity information of the two sequences (or to filtered versions of them), without prior segmentation of foreground moving objects, and without prior detection of corresponding features across the sequences. Our approach offers two main advantages over existing approaches to sequence alignment:
1. It is capable of aligning sequences of the same *action* performed at different times and places by different people wearing different clothes, regardless of their photometric properties and other static or dynamic scene clutter.
2. It provides a unified approach to multi-sensor sequence alignment for a wide range of scenarios, including: (i) sequences taken with either stationary or jointly moving cameras, (ii) sequences with the same or different photometric properties, and (iii) sequences with or without moving objects. Our approach does assume, however, that the cameras are rigid with respect to each other (although they may move jointly).

The remainder of this work is organized as follows: Sec. 2 formulates the problem, Sec. 3 presents the space-time similarity measure between the two sequences. Sec. 4 presents the space-time alignment algorithm. Sec. 5 provides experimental results on real sequences. Sec. 6 discusses the robustness of the algorithm to noise and to other dynamic scene clutter.

2 Problem Formulation

Given two sequences, f and g, we seek the spatio-temporal parametric transformation $\boldsymbol{p}$ that maximizes a global similarity measure M between the two sequences after bringing them into alignment according to $\boldsymbol{p}$. f and g may be either the original video sequences, or some filtered version of them, depending on the underlying application (see Sec. 5).

For each space-time point (x, y, t) in the sequence f, we denote its spatio-temporal displacement vector by $\boldsymbol{u} = (u_1, u_2, u_3)$. $\boldsymbol{u}$ is a function of both the space-time point coordinates and the unknown parameter vector $\boldsymbol{p}$, i.e., $\boldsymbol{u} = \boldsymbol{u}(x, y, t; \boldsymbol{p})$.

We assume that the *relative* internal and external parameters between the cameras are fixed (but unknown). The cameras may be either stationary or moving (jointly). In our current implementation we have chosen a 2D affine transformation to model the *spatial transformation* between corresponding frames across the two sequences (such a model is applicable when the scene is planar, or distant, or when the two cameras are relatively close to each other). A 1D affine transformation was chosen to model the *temporal transformation* between the two sequences (supporting sequences with different frame rates as well as a time offset between the sequences). The space-time transformation $\boldsymbol{p}$ therefore comprises of 8 parameters, where the first 6 parameters $(p_1, \ldots, p_6)$ capture the spatial 2D affine transformation and the remaining 2 parameters (p_7, p_8) capture the temporal 1D affine transformation. The spatio-temporal displacement vector $\boldsymbol{u}(x, y, t; \boldsymbol{p})$ is therefore:

$$\boldsymbol{u}(x, y, t; \boldsymbol{p}) = \begin{bmatrix} u_1(x, y, t; \boldsymbol{p}) \\ u_2(x, y, t; \boldsymbol{p}) \\ u_3(x, y, t; \boldsymbol{p}) \end{bmatrix} = \begin{bmatrix} p_1 x + p_2 y + p_3 \\ p_4 x + p_5 y + p_6 \\ p_7 t + p_8 \end{bmatrix}$$

This can be written more compactly as:

$$\boldsymbol{u}(x, y, t; \boldsymbol{p}) = X(x, y, t) \cdot \boldsymbol{p} \qquad (1)$$

where $\boldsymbol{p} = (p_1, \ldots, p_8)$, and:

$$X(x, y, t) = \begin{bmatrix} x\ y\ 1\ 0\ 0\ 0\ 0\ 0 \\ 0\ 0\ 0\ x\ y\ 1\ 0\ 0 \\ 0\ 0\ 0\ 0\ 0\ 0\ t\ 1 \end{bmatrix} .$$

3 The Similarity Measure

Sequences of actions recorded at different times and places, as well as sequences obtained by different sensing modalities (e.g., an IR and a visible light camera) have significantly different photometric properties. As such, their intensities are related by highly non-linear transformations.

In [8] the following observations were made for a multi-sensor *image* pair: (i) the intensities of images taken with sensors of different modalities are usually related by a highly non-linear global transformation which depends not only on the image intensity, but also on its image location. Such intensity transformations are not handled well by Mutual Information (which assumes spatial invariance). Nevertheless, (ii) for very small corresponding image patches across the two images, their intensities are *locally* related by some *linear* intensity transformation. Since normalized-correlation is invariant to linear intensity transformations, it can be used as a *local* similarity measure applied to small image patches.

Our approach is based on extending this approach to space-time, and takes it beyond multi-sensor alignment, to alignment of actions. Local normalized correlations are computed within small *space-time patches* (in our implementation they were of size $7 \times 7 \times 7$). A *global* similarity measure is then computed as the sum of all those local measures in the entire sequence. The resulting global similarity measure is thus invariant to spatially and temporally varying non-linear intensity transformations.

Given two corresponding space-time patches/windows, w_f and w_g, one from each sequence, their local Normalized Correlation (NC) can be estimated as follows [7]: $NC(w_f, w_g) = \frac{\text{cov}(w_f,w_g)}{\sqrt{\text{var}(w_f)}\sqrt{\text{var}(w_g)}}$, where cov and var stand for the covariance and variance of intensities. Squaring the NC measure further accounts for contrast reversal, which is common in multi-sensor sequence pairs. Our patch-wise local similarity measure is therefore:

$$C(w_f, w_g) = \frac{\text{cov}^2(w_f, w_g)}{\text{var}(w_f)\text{var}(w_g) + \alpha} \tag{2}$$

where the constant α is added to account for noise (in our experiments we used $\alpha = 10$, but the algorithm is not particularly sensitive to the choice of α).

The *global* similarity measure M between two sequences (f and g) is computed as the sum of all the *local* measures C applied to small space-time patches around each pixel in the sequence:

$$M(f, g) = \sum_x \sum_y \sum_t C\left(w_f(x, y, t), w_g(x, y, t)\right) \tag{3}$$

This results in a global measure which is invariant to highly non-linear intensity transformations (which may vary spatially and temporally over the sequences).

Our goal is to recover the global geometric space-time transformation which maximizes the global measure M between the two sequences. To do so, we reformulate the local measure C and the global measure M in terms of the unknown parametric transformation $\boldsymbol{p}$. For each space-time point (x, y, t) in the sequence f and its spatio-temporal displacement vector $\boldsymbol{u} = (u_1, u_2, u_3)$, the local normalized correlation measure of Eq. (2) can be written as a function of $\boldsymbol{u}$:

$$C^{(x,y,t)}(\boldsymbol{u}) = C\left(w_f(x, y, t), w_g(x + u_1, y + u_2, t + u_3)\right)$$

where $w_f(x, y, t)$ is the $7 \times 7 \times 7$ space-time window around pixel (x, y, t) in f, and $w_g(x+u_1, y+u_2, t+u_3)$ is the $7 \times 7 \times 7$ space-time window around pixel $(x+u_1, y+u_2, t+u_3)$ in g. We can therefore formulate the alignment problem as follows: Find $\boldsymbol{p}$ (the set of global spatio-temporal parameters) that maximizes the global similarity measure $M(\boldsymbol{p})$:

$$M(\boldsymbol{p}) = \sum_{(x,y,t) \in f} C^{(x,y,t)}\left(\boldsymbol{u}(x, y, t; \boldsymbol{p})\right) \tag{4}$$

4 The Alignment Algorithm

4.1 The Maximization Process

We use Newton's method [9] for the optimization task. Local quadratic approximations of $M(\boldsymbol{p})$ are used in order to iteratively converge to the correct value of the space-time transformation $\boldsymbol{p}$. Let $\boldsymbol{p}_0$ be the current estimate of the transformation parameters $\boldsymbol{p}$. We can write the quadratic approximation of $M(\boldsymbol{p})$ around $\boldsymbol{p}_0$ as:

$$M(\boldsymbol{p}) = M(\boldsymbol{p}_0) + \left(\nabla_{\boldsymbol{p}} M(\boldsymbol{p}_0)\right)^T \boldsymbol{\delta}_p + \frac{1}{2}\boldsymbol{\delta}_p^T H_M(\boldsymbol{p}_0)\boldsymbol{\delta}_p$$

where $\nabla_{\boldsymbol{p}} M$ and H_M are the gradient and hessian of M, respectively (both computed around $\boldsymbol{p}_0$), and $\boldsymbol{\delta}_p = \boldsymbol{p} - \boldsymbol{p}_0$ is the unknown refinement step. By differentiating this approximation with respect to $\boldsymbol{\delta}_p$ and equating to zero, we obtain the following expression for $\boldsymbol{\delta}_p$:

$$\boldsymbol{\delta}_p = -\left(H_M(\boldsymbol{p}_0)\right)^{-1} \cdot \nabla_{\boldsymbol{p}} M(\boldsymbol{p}_0) \tag{5}$$

From Eqs. (1) and (4) and the chain rule of differentiation, we can evaluate $\nabla_{\boldsymbol{p}} M$ and H_M:

$$\begin{aligned} \nabla_{\boldsymbol{p}} M(\boldsymbol{p}) &= \sum_{(x,y,t)\in f} \nabla_{\boldsymbol{p}} C^{(x,y,t)}(\boldsymbol{u}) \\ &= \sum_{(x,y,t)\in f} \left(X^T \cdot \nabla_{\boldsymbol{u}} C^{(x,y,t)}(\boldsymbol{u})\right) \qquad (6) \\ H_M(\boldsymbol{p}) &= \sum_{(x,y,t)\in f} \left(X^T \cdot H_{C^{(x,y,t)}(\boldsymbol{u})} \cdot X\right) \qquad (7) \end{aligned}$$

where $\nabla_{\boldsymbol{u}} C^{(x,y,t)}$ and $H_{C^{(x,y,t)}}$ are the gradient and hessian of $C^{(x,y,t)}(\boldsymbol{u})$, respectively, computed around $\boldsymbol{u}_0 = \boldsymbol{u}(x, y, t; \boldsymbol{p}_0)$. Substituting Eq. (6) and Eq. (7) into Eq. (5), we get the following expression for the refinement step $\boldsymbol{\delta}_p$, in terms of the normalized correlation function $C^{(x,y,t)}(\boldsymbol{u})$:

$$\boldsymbol{\delta}_p = -\left(\sum_{(x,y,t)\in f} X^T H_{C^{(x,y,t)}(\boldsymbol{u}_0)} X\right)^{-1} \cdot \sum_{(x,y,t)\in f} X^T \nabla_{\boldsymbol{u}} C^{(x,y,t)}(\boldsymbol{u}_0) \tag{8}$$

In order to calculate the refinement step of Eq. (8) we need to differentiate the normalized correlation function $C^{(x,y,t)}$ of each space-time point (x, y, t) around its currently estimated displacement vector $\boldsymbol{u}_0 = \boldsymbol{u}(x, y, t; \boldsymbol{p}_0)$. This is done as follows: For each space-time point (x, y, t), a local normalized correlation function (volume) $C^{(x,y,t)}(\boldsymbol{u})$ is evaluated for a set of spatio-temporal displacements around $\boldsymbol{u}_0$. Then, the first and second derivatives of $C^{(x,y,t)}$ with respect to $\boldsymbol{u} = (u_1, u_2, u_3)$ are extracted in order to obtain $\nabla_{\boldsymbol{u}} C^{(x,y,t)}$ and $H_{C^{(x,y,t)}}$:

$$\nabla_{\boldsymbol{u}} C^{(x,y,t)} = \left[\frac{\partial C^{(x,y,t)}}{\partial x} \; \frac{\partial C^{(x,y,t)}}{\partial y} \; \frac{\partial C^{(x,y,t)}}{\partial t}\right]^T$$

$$H_{C^{(x,y,t)}} = \begin{bmatrix} \frac{\partial^2 C^{(x,y,t)}}{\partial x^2} & \frac{\partial^2 C^{(x,y,t)}}{\partial x \partial y} & \frac{\partial^2 C^{(x,y,t)}}{\partial x \partial t} \\ \frac{\partial^2 C^{(x,y,t)}}{\partial y \partial x} & \frac{\partial^2 C^{(x,y,t)}}{\partial y^2} & \frac{\partial^2 C^{(x,y,t)}}{\partial y \partial t} \\ \frac{\partial^2 C^{(x,y,t)}}{\partial t \partial x} & \frac{\partial^2 C^{(x,y,t)}}{\partial t \partial y} & \frac{\partial^2 C^{(x,y,t)}}{\partial t^2} \end{bmatrix} \tag{9}$$

In practice, we evaluate $C^{(x,y,t)}(\boldsymbol{u})$ for displacements of $\boldsymbol{u}_0 \pm 2$ in x, y, t (i.e., the correlation function is a volume of size $5 \times 5 \times 5$). To account for large misalignments, the above maximization scheme is performed within a coarse-to-fine data structure. The resulting algorithm is therefore as follows:

The Algorithm:

1. Construct a space-time Gaussian pyramid for each sequence (Sec. 4.3).

2. Find an initial guess $\boldsymbol{p}_0$ for the space-time transformation parameters in the coarsest (smallest) pyramid level (Sec. 4.4).

3. Apply several maximization iterations in the current pyramid level until convergence. In each iteration do:

(a) Use the current parameter estimate $\boldsymbol{p}_0$ from the last iteration to compute the refinement step $\boldsymbol{\delta}_p$ (Eq. (8)).

(b) Update the current parameter estimate $\boldsymbol{p}_0 = \boldsymbol{p}_0 + \boldsymbol{\delta}_p$.

(c) Test for convergence: If the change in the values of $M(\boldsymbol{p})$ for two successive iterations is small enough, go to step 4. Otherwise, go back to step 3.(a).

4. Proceed to the next pyramid level and go back to step 3.

4.2 Confidence-Weighted Regression

To further stabilize the maximization process, we consider only space-time points (x, y, t) in which the quadratic approximation of the normalized correlation function is *concave*. Other space-time points are ignored (are outliers), since they incorporate false information into the regression. Moreover, the contribution of each space-time point is weighted by its reliability, which is measured by the degree of concavity of the normalized correlation function at this point.

A twice-differentiable function is concave at a point if and only if the hessian of the function at the point is negative semidefinite [12], i.e., if all its k^{th} order leading principal minors are non-positive for an odd k and non-negative for an even k. Therefore, the hessian matrix $H_{C^{(x,y,t)}(\boldsymbol{u}_0)}$ of Eq. (9) is checked for negative semidefiniteness by:

$$\left|H_{C(\boldsymbol{u}_0)}\right| \leq 0 \ , \quad \begin{vmatrix} \frac{\partial^2 C(\boldsymbol{u}_0)}{\partial x^2} & \frac{\partial^2 C(\boldsymbol{u}_0)}{\partial x \partial y} \\ \frac{\partial^2 C(\boldsymbol{u}_0)}{\partial y \partial x} & \frac{\partial^2 C(\boldsymbol{u}_0)}{\partial y^2} \end{vmatrix} \geq 0 \ , \quad \frac{\partial^2 C(\boldsymbol{u}_0)}{\partial x^2} \leq 0$$

where $C(\boldsymbol{u}_0) = C^{(x,y,t)}(\boldsymbol{u}_0)$, and $|\cdot|$ denotes the determinant of a matrix. Only space-time points (x, y, t) in which the corresponding hessian $H_{C^{(x,y,t)}(\boldsymbol{u}_0)}$ is negative semidefinite are considered as inliers in the maximization process. Let S denote this set of inlier space-time points. Each space-time point in S is further weighted by the determinant of its corresponding hessian, which indicates the degree of concavity at that point. This outlier rejection and weighting scheme is incorporated into the algorithm by extending Eq. (8):

$$\boldsymbol{\delta}_p = -\left(\sum_{(x,y,t)\in S} w(\boldsymbol{u}_0) X^T H_{C(\boldsymbol{u}_0)} X\right)^{-1} \cdot \sum_{(x,y,t)\in S} w(\boldsymbol{u}_0) X^T \nabla_{\boldsymbol{u}} C(\boldsymbol{u}_0)$$

where $w(\boldsymbol{u}_0) = w^{(x,y,t)}(\boldsymbol{u}_0) = -\left|H_{C(\boldsymbol{u}_0)}\right|$.

4.3 The Space-Time Gaussian Pyramid

To handle large spatio-temporal misalignments between the two sequences, the optimization is done coarse-to-fine (in space and in time). Caspi and Irani [4] presented a space-time Gaussian pyramid for video sequences. Each pyramid level was constructed by applying a Gaussian low-pass filter to the previous level, followed by sub-sampling by a factor of 2. The filtering and sub-sampling phases were performed both in space and in time (i.e., in x, y and t). Our coarse-to-fine estimation is performed within such a data structure, with a small modification to handle sequences whose temporal and spatial dimensions are significantly different (otherwise, the coarsest pyramid level will be too coarse in one dimension, while not coarse enough in the other dimensions). Filtering and sub-sampling is first applied along the largest dimension(s), until it is of similar size to the other dimensions, and then proceeding as in [4]. To guarantee numerical stability, the coarsest (smallest) pyramid level is at least $30 \times 30 \times 30$.

4.4 The Initial Parametric Transformation $\boldsymbol{p}_0$

An initial guess $\boldsymbol{p}_0$ for the space-time parametric transformation is computed at the coarsest pyramid level. We seek for initial non-zero values only for the translational parameters of $\boldsymbol{p}$ in x, y and t (i.e., p_3, p_6 and p_8), leaving all the other parameters to be zero. This is done by evaluating the similarity measure M of Eq. (4) for each possible spatio-temporal integer shift within a search radius (in our implementation we used a radius of 25% of the sequence in each dimension). The translation parameters that provide the highest similarity value M are used in the initial guess $\boldsymbol{p}_0$ for the transformation parameters. Initializing only p_3, p_6 and p_8 is usually sufficient for the initial guess. The remaining parameters in $\boldsymbol{p}$ tend to be smaller, and initializing them with zero-values usually suffice for convergence. All the parameters in $\boldsymbol{p}$ are updated during the optimization process. Note that although an "exhaustive" search is performed at the coarsest pyramid level, this process is *not* time consuming since the smallest spatio-temporal pyramid level is typically of size $30 \times 30 \times 30$.

5 Applications and Results

Recall that we focus on two applications of sequence alignment: (1) Alignment of action sequences, taken at different times and places, and (2) Alignment of sequences recorded simultaneously by different cameras, where the most difficult case is when these are

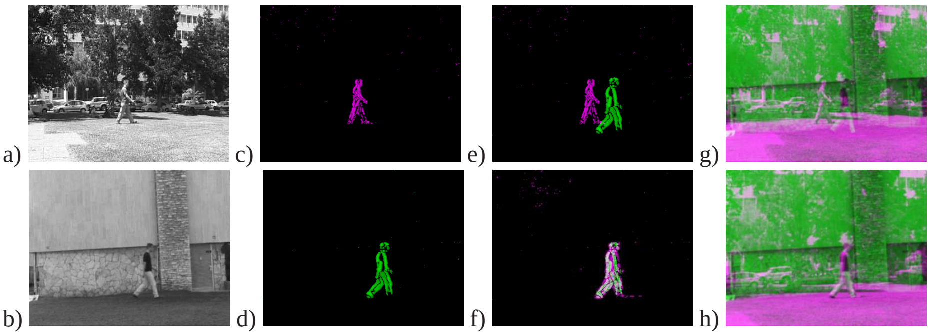

Fig. 1. Action alignment. (a) and (b) show frames 74 of the two input sequences, f and g. (c) and (d) show the absolute value of their temporal derivatives (f_t^{abs} in magenta and g_t^{abs} in green). (e) Initial misalignment (superposition of (c) and (d)). (f) Superposition of corresponding frames after alignment both in space and in time. The white color is a result of superposition of the green and magenta. (g) and (h) show superposition of the input sequences before and after alignment (one in green and one in magenta). **For color figure and full video sequence see http://www.wisdom.weizmann.ac.il/∼vision/SpaceTimeCorrelations.html.**

sensors of different modalities. We use the same alignment algorithm for these two applications. However, we apply the algorithm to different sequence *representations*, which are obtained by pre-filtering the original input sequences with different linear filters. These prior filters emphasize the part of the data which we want to bring into alignment. The chosen filters for each application along with experimental results are presented next.

5.1 Multi-Sensor Alignment

The common information across a multi-sensor pair of sequences (e.g., infra-red and visible-light) is the *details* in the scene (spatial or temporal). These are captured mostly by high-frequency information (both in time and in space). The multi-sensor pair differ in their photometric properties which are captured by low frequencies. Thus, to enhance the common detail information and suppress the non-common photometric properties, differentiation operators are applied to the sequences. Since directional information is important, the input sequences f and g are differentiated separately with respect to x, y and t, resulting in three sequences of directional derivatives (f_x, f_y, f_t and g_x, g_y, g_t). An absolute value is further taken to account for contrast reversal. Thus, the global similarity measure of Eq. (3) becomes:

$$M(f,g) = M\big(f_x^{abs}, g_x^{abs}\big) + M\big(f_y^{abs}, g_y^{abs}\big) + M\big(f_t^{abs}, g_t^{abs}\big)$$

Due to lack of space we omitted the figures of the multi-sensor alignment results from the paper. However, these results (i.e., multi-sensor sequences before and after space-time alignment) can be found on our web site:
http://www.wisdom.weizmann.ac.il/∼vision/SpaceTimeCorrelations.html. We display there different examples of multi-sensor pairs obtained under different scenarios – in

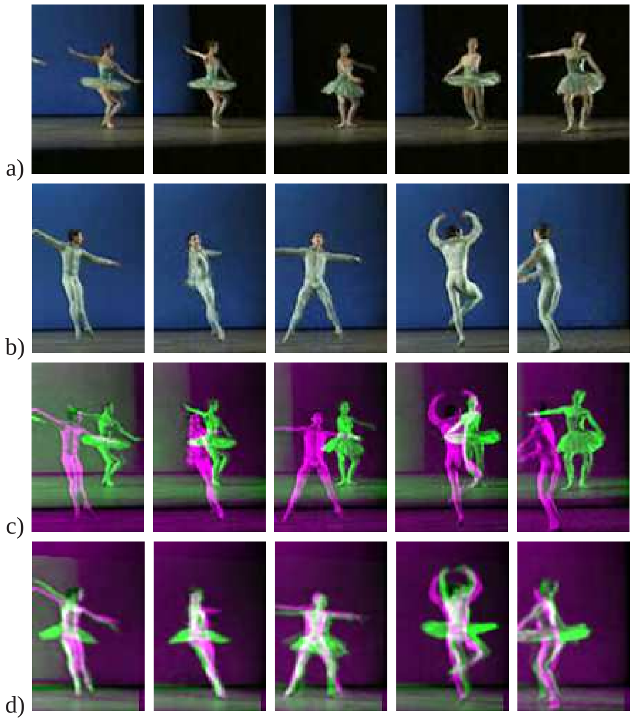

Fig. 2. Action alignment. (a) and (b) show several frames of the two input sequences, f and g (with same frame numbers). (c) shows superposition of (a) and (b) before alignment (f in green and g in magenta). (d) shows superposition of corresponding frames after alignment both in space and in time. This compensates for the *global parametric* geometric deformations (spatial scale, speed, orientation, position, etc.) The residual *non-parametric local* deformations highlight the differences in performance of the two dancers. **For color figure and full video sequence see http://www.wisdom.weizmann.ac.il/∼vision/SpaceTimeCorrelations.html.**

one case the cameras are moving, while in another case the cameras are still and there are moving objects in the scene. All these sequence pairs were brought into space-time alignment using the above algorithm. Previous methods for sequence alignment were usually restricted to one type of scenario (either moving cameras [3] or moving objects [5]).

5.2 Action Alignment

Given two sequences that contain a similar action, performed by different people at different times and places, we would like to align only the action (i.e., the foreground moving objects), ignoring the different backgrounds and the photometric properties of the sequences. For example, given two sequences of walking people, we want to align only the walking people themselves, regardless of their backgrounds, the scale and orientation of the walking people, the walking speed, the illumination, and the clothing

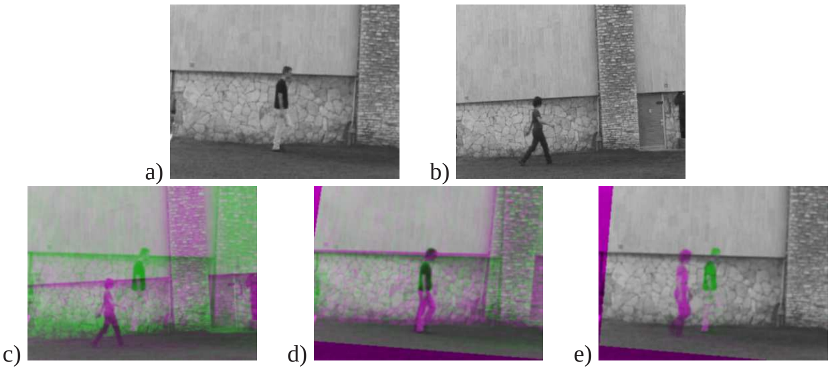

Fig. 3. Action alignment vs. background alignment. (a) and (b) show frame 45 of the two input sequences. (c) Initial misalignment (superposition of (a) and (b)). (d) Superposition after space-time alignment using temporal derivatives only (Eq. (10)). (e) Superposition after space-time alignment using spatial derivatives only (Eq. (11)). **For color figure and full video sequence see http://www.wisdom.weizmann.ac.il/$\sim$vision/SpaceTimeCorrelations.html.**

colors. The common information in two such sequences is captured mostly by the temporal variations (derivatives), and not by the spatial ones. Therefore, for the purpose of Action Alignment, Eq. (3) becomes:

$$M(f,g) = M\left(f_t^{abs}, g_t^{abs}\right) \tag{10}$$

The two sequences in Fig. 1.a and 1.b contain a person walking at different times and in different places (the cameras are stationary). There are four significant differences between the two input sequences: (1) their backgrounds are different (trees in one sequence, and a wall in the other), (2) the spatial scale of the walking person is significantly different (by approximately 36%), (3) the walking speed is different (be approximately 13%), and (4) the clothing colors are different. Figs. 1.c and 1.d show the absolute values of the temporal derivatives (f_t^{abs} and g_t^{abs}) of the input sequences. Fig. 1.e displays the initial misalignment between the two sequences through an overlay of 1.c and 1.d before alignment. Fig. 1.f shows the same display after alignment of the actions both in space and in time. The white color in Fig. 1.f is obtained from superposition of the green and magenta, which indicates good alignment (please see color figures and color sequences on our web site). Figs 1.g and 1.h display super-position of the two input video sequences before and after alignment, respectively (where one sequence is displayed in green and the other sequence is displayed in magenta).

The two sequences in Fig. 2.a and 2.b contain two different dancers that perform a similar ballet dance. Figs 2.c display super-position of the two input video sequences before alignment (where the first sequence is displayed in green and the second sequence is displayed in magenta). Initially, the two dancers are misaligned both in space and in time. 2.d shows a similar super-position after alignment. The two dancers are now aligned both in space and in time (although their movements are not identical).

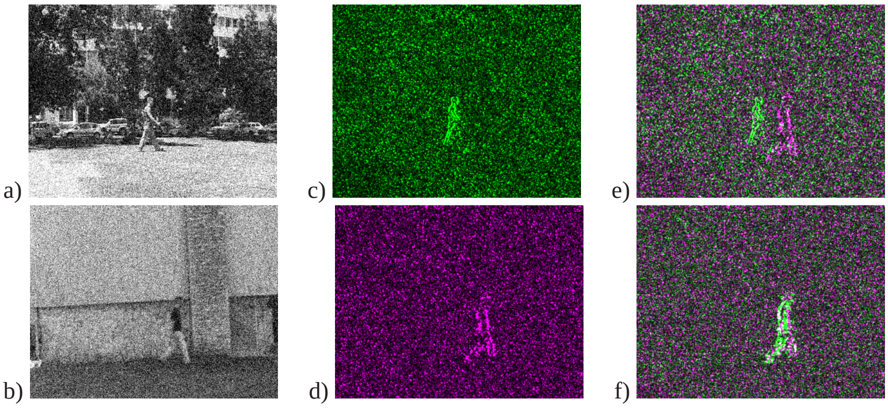

Fig. 4. Robustness to noise. (a) and (b) show frame 74 of the two noisy input sequences (see text for more details). (c) and (d) show the absolute value of the temporal derivatives of (a) and (b), respectively. (e) Initial misalignment (superposition of (c) and (d)). (f) Superposition after alignment in space and in time. **For color figure and full video sequence see http://www.wisdom.weizmann.ac.il/∼vision/SpaceTimeCorrelations.html.**

Applications: This capability of aligning actions can be used for various applications, including: (i) Action/Event recognition: Given a sequence of an action and a database of sequences with different actions, find the action in the database that achieves best alignment with the query action, i.e., that yields the highest value for the measure M of Eq. (4). (ii) Identification of people by the way they behave: Given a sequence of a person performing some action, and a database of different people performing the same action, find the database sequence that provides the best alignment (maximal score M) with the query sequence. This will allow to identify the person in the query sequence. Carlsson [2] proposed an algorithm for recognizing people by the way they walk. However, his algorithm required manual marking of specific body locations in each frame of the two sequences, whereas our approach is automatic. (iii) Comparing performance and style of people in various sport activities.

Action Alignment vs. **Background Alignment:** The choice of the sequence representation is important. For example, consider the two input sequences in Fig. 3.a and 3.b. There are two different people walking against the same background (recorded at different times). Fig. 3.c shows the initial misalignment between the two input sequences. Note that both the walking people and their backgrounds are not aligned. Fig. 3.d shows the results of applying the alignment algorithm to the derivatives of the input sequences with respect to t alone (using the global similarity measure in Eq. (10)). As expected, only the actions are aligned, and the backgrounds are not aligned. Figure 3.e shows the results of applying the alignment algorithm to the derivatives of the same input sequences, but this time differentiated with respect to x and y. This is done by replacing Eq. (3) with:

$$M(f,g) = M\left(f_x^{abs}, g_x^{abs}\right) + M\left(f_y^{abs}, g_y^{abs}\right) \tag{11}$$

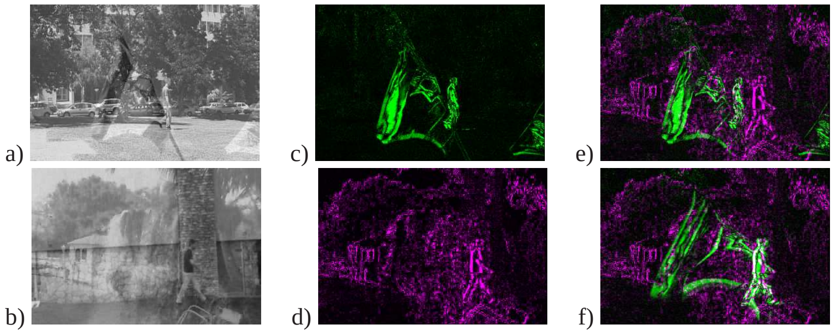

Fig. 5. The locking property. (a) Frame 61 of the first sequence: a mixture of the sequence of Fig. 1.a with a flag sequence. (b) Frame 61 of the second sequence: a mixture of the sequence of Fig. 1.b with a waterfall sequence. (c) and (d) show the absolute value of the temporal derivatives of (a) and (b), respectively. (e) Initial misalignment (superposition of (c) and (d)). (f) Superposition after alignment in space and in time. The algorithm locks onto the common walking action, despite the presence of other scene dynamics. **For color figure and full video sequence see http://www.wisdom.weizmann.ac.il/∼vision/SpaceTimeCorrelations.html.**

Since only the spatial variations of the sequences are used in the alignment process, the backgrounds are brought into alignment, while the walking people are not.

6 Robustness & Locking Property

One of the benefits of a coarse-to-fine estimation process is the “locking property”, which provides robustness to noise, as well as the ability to lock onto a dominant space-time transformation. Burt *et al.* [1] discussed this effect in the context of *image alignment* in the presence of multiple motions. According to [1], since pyramids provide a separation of the spectrum into different frequency bands, motion components with different frequency characteristics tend to be separated. This separation causes the motion estimator to “lock” onto a single (dominant) motion component, even when other motions are present. A similar phenomena occurs in our sequence alignment algorithm, which tends to lock onto a *dominant space-time coordinate transformation* between the two sequences. Figs. 4 and 5 demonstrate the locking property.

Fig. 4 displays the robustness of our algorithm to noise. Gaussian noise with zero mean and a standard deviation of 40 gray-level units (out of 255) was added to the two input sequences of Fig. 1.a and 1.b. The resulting sequences are shown in Figs. 4.a and 4.b. Figs. 4.c and 4.d display the absolute values of the temporal derivatives of the input sequences. The presence of a significant noise is clearly seen in these figures. An overlay of 4.c and 4.d before alignment is shown in Fig. 4.e. Fig. 4.f displays an overlay of corresponding frames after alignment in space and in time. Good alignment is obtained despite the significant noise.

Fig. 5 displays the locking property in the case of multiple transparent layers. Again, we took the two input sequences of Fig. 1.a and 1.b, but this time mixed them with two different sequences that contain significant non-rigid motions (a waving flag and a waterfall). The first input sequence (Fig. 5.a) contains a walking person (with trees in the background) mixed with a waving flag, and the second input sequence (Fig. 5.b) contains a walking person (with a wall in the background) mixed with a waterfall. Figs. 5.c and 5.d display the absolute values of the temporal derivatives of the input sequences. The presence of the multiple layers is clearly seen in these figures. An overlay of 5.c and 5.d before alignment is shown in Fig. 5.e. Fig. 5.f displays an overlay of corresponding frames after alignment in space and in time. The white color in Fig. 5.f indicates that the algorithm automatically locked on the common walking action, despite the other scene dynamics. The regression was applied to the entire sequence. This illustrates the strong locking property of the algorithm. The results can be seen much more clearly in the video on our web site.

7 Summary

We introduced an algorithm for sequence alignment, based on maximizing local space-time correlations. Our algorithm aligns sequences of the same action performed at different times and places by different people, possibly at different speeds, and wearing different clothes. Moreover, the algorithm offers a unified approach to the sequence alignment problem for a wide range of scenarios (sequence pairs taken with stationary or jointly moving cameras, with the same or different photometric properties, with or without moving objects). Our algorithm is applied directly to the dense space-time intensity information of the two sequences (or to filtered versions of them). This is done without prior segmentation of foreground moving objects, and without prior detection of corresponding features across the sequences.

References

1. P. Burt, R. Hingorani, and R. Kolczynski, “Mechanisms for isolating component patterns in the sequential analysis of multiple motion,” in *Workshop on Visual Motion*, 1991.
2. S. Carlsson, “Recognizing walking people,” in *IJRR*, vol. 22, pp. 359–370, 2003.
3. Y. Caspi and M. Irani, “Aligning non-overlapping sequences,” *IJCV*, vol. 48, 2002.
4. Y. Caspi and M. Irani, “Spatio-temporal alignment of sequences,” *T-PAMI*, 2002.
5. Y. Caspi, D. Simakov, and M. Irani, “Feature-based sequence-to-sequence matching,” in *VMODS*, 2002.
6. M. A. Giese and T. Poggio, “Synthesis and recognition of biological motion patterns based on linear superposition of prototypical motion sequences,” in *IEEE Workshop on Multi-View Modeling and Analysis of Visual Scenes*, 1999.
7. R. C. Gonzalez and R. E. Woods, *Digital Image Processing*. Addison-Wesley, 1993.
8. M. Irani and P. Anandan, “Robust multi-sensor image alignment,” in *ICCV*, 1998.
9. W. Press, B. Flannery, S. Teukolsky, and W. Vetterling, *Numerical Recipes in C*. Cambridge Univ. Press, 1988.
10. C. Rao, A. Gritai, M. Shah, and T. Syeda-Mahmood, “View-invariant alignment and matching of video sequences,” in *ICCV*, pp. 939–945, 2004.
11. E. Shechtman and M. Irani, “Space-time behavior based correlation,” in *CVPR*, 2005.
12. G. Thomas and R. Finney, *Calculus and Analytic Geometry (9th Edition)*. Addison-Wesley, 1996.