Video Editing with Temporal, Spatial and Appearance Consistency

Xiaojie Guo[1], Xiaochun Cao[1,2], Xiaowu Chen[3] and Yi Ma[4]
[1]School of Computer Science and Technology, Tianjin University, Tianjin, 300072, China
[2]State Key Laboratory Of Information Security, IIE, CAS, Beijing, 100093, China
[3]School of Computer Science and Engineering, Beihang University, Beijing, 100191, China
[4]Visual Computing Group, Microsoft Research Asia, Beijing, 100080, China
xguo@tju.edu.cn, caoxiaochun@iie.ac.cn, chen@vrlab.buaa.edu.cn, mayi@microsoft.com

Abstract

Given an area of interest in a video sequence, one may want to manipulate or edit the area, e.g. remove occlusions from or replace with an advertisement on it. Such a task involves three main challenges including temporal consistency, spatial pose, and visual realism. The proposed method effectively seeks an optimal solution to simultaneously deal with temporal alignment, pose rectification, as well as precise recovery of the occlusion. To make our method applicable to long video sequences, we propose a batch alignment method for automatically aligning and rectifying a small number of initial frames, and then show how to align the remaining frames incrementally to the aligned base images. From the error residual of the robust alignment process, we automatically construct a trimap of the region for each frame, which is used as the input to alpha matting methods to extract the occluding foreground. Experimental results on both simulated and real data demonstrate the accurate and robust performance of our method.

1. Introduction

There exist many tools to edit an image to meet different demands, such as removing, adding or replacing targets [13][14], inpainting [11] and color manipulation [6]. For instance, if one wants to change the original facade bounded by the green window (Fig. 1 left) into the target in the right, two operations are required: registering the target to the source, and separating foreground. Some of the existing tools already allow the users to process one or two such images interactively. But editing a long sequence of images remains extremely difficult. Since the sequence is usually captured by a hand-held camera, images of the region of interest will appear to be scaled, rotated or deformed throughout the sequence. Thus, the consistency of editing results across the image sequence and the amount of manual interaction are two extra issues need to be considered.

Figure 1. An example of video editing. **Left:** an original frame, where the facade of a selected building and the occlusion map are shown within the small windows. **Right:** the result by replacing the facade with a new texture. The three small windows are the new facade (top-left), the trimap (bottom-left), and the mask of occlusion (bottom-right), respectively.

To guarantee visual consistency and alleviate human interaction, we first need to automatically align the region of interest precisely across the sequence. [8] and [7] propose to align images by minimizing the sum of entropies of pixel values at each pixel location in the batch of aligned images. Conversely, the least squares congealing procedure of [3], [4] seeks an alignment that minimizes the sum of squared distances between pairs of images. Vedaldi *et al.* [16] minimize a log-determinant measure to accomplish the task. The major drawback of the above approaches is that they do not simultaneously handle large illumination variations and gross pixel corruptions or partial occlusions that usually occur in real images. To overcome the drawback, Peng *et al.* [12] propose an algorithm named RASL to solve the task of image alignment by low-rank and sparse decomposition. This method takes into account gross corruptions, and thus gives better results in real applications.

Spatial pose (or 2D deformation of the region) also affects the visual quality of editing. RASL only takes care of temporal alignment but not the 2D deformation of the region being aligned. As can be seen from Fig. 3 (d), the

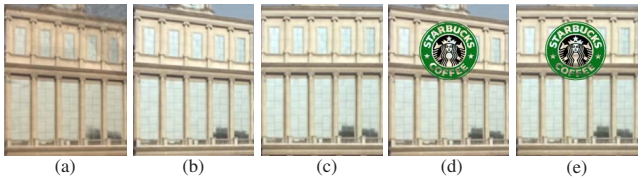
Figure 2. Some real world examples. The rectified versions of the objects (bounded by red dotted windows) are displayed on top-left.

circular Starbucks logo looks awkward if directly pasted on the deformed facade. One way of improving visual quality of editing is to transform the logo according to the facade pose. But this is inconvenient for users to do manually. Another way is to rectify the facade as shown in Fig. 3 (e). This can be done by TILT [19], which assumes that the rectified texture has lower rank than its deformed versions. Actually, it is pervasive that many natural and man-made things (of interest for editing) have low-rank appearance in their rectified position. Some examples are shown in Fig. 2, in which the rank of each rectified (top-left) image is much lower than that of its original. In this paper, we adopt the same assumption as [19]. However, TILT alone is not adequate for our task, since TILT is insensitive to translation and scaling of the texture (see [19] for reasons). As a result, applying TILT independently on each frame is very likely to introduce misalignment. Both RASL and TILT are based on the common mathematical framework of Robust PCA [1]. Hence it is natural to try to combine them together to handle the alignment and rectification problem for video editing.

Besides, to make the editing results look natural and real, it is crucial to preserve the occluding foreground. That means the foreground need to be precisely matted, extracted, and synthesized too. Plenty of literature about alpha matting has been published [9][2][15]. But these methods require human interaction for initializing trimap. In this work, we will show how this can be done automatically.

As discussed above, temporal alignment, spatial rectification, and occlusion of the area are three main issues we have to handle properly for the desired video editing task. In this paper, we propose an effective and efficient framework that simultaneously handles these three challenges. The core of the proposed framework is based on sparse and low-rank decomposition. With the consideration of the limitation of sparse and low-rank decomposition on large data, we first construct the area basis based on several images. Every individual area is then parsed into its intrinsic appearance and the corresponding residual, according to the well-constructed basis. We automatically generate the trimap from the area and its residual, then adopt alpha matting methods to extract the foreground.

Contributions of this paper: 1) Our work is the first to consider both temporal and spatial image correlation in a unified framework. 2) We provide an approach to applying the low-rank and sparse decomposition techniques on large datasets, significantly broaden the applicable range of RPCA-based methods. 3) This paper gives a new way to edit a target (with low-rank textures) in videos with minimal human intervention.



(a) (b) (c) (d) (e)

Figure 3. An example of image alignment and rectification. (a) The median of initial inputs without either alignment or rectification. (b) The median after alignment by RASL only. (c) The median with both alignment and rectification by our method. (d) and (e) Adding a circular logo on (b) and (c), respectively.

2. Our Method

2.1. Initial Batch Frame Alignment

In the simplest case, the areas of interest are clean (without occlusions or gross corruptions) and temporally well-aligned. If we stack the 2D areas as columns of a matrix $\boldsymbol{B} \in \mathbb{R}^{m \times n}$, where m is the number of pixels of the area, the matrix $\boldsymbol{B}$ is low rank. In the real world, the areas are very likely to have partial occlusions or specular reflections, which may break the low rankness of the matrix $\boldsymbol{B}$. Since these errors typically affect only a small fraction of an area, we treat them as sparse errors whose nonzero entries can have arbitrarily large magnitudes. Therefore, we have $\boldsymbol{B} + \boldsymbol{E} = \boldsymbol{A}$, where $\boldsymbol{E}$ is the sparse residual matrix between the corrupted matrix $\boldsymbol{A}$ and the clean (low rank) area matrix $\boldsymbol{B}$. As aforementioned, the areas are probably temporally misaligned. In this work, we assume that the area lies on a (nearly) planar surface in the scene. Let $\boldsymbol{A}_1$, $\boldsymbol{A}_2$,..., $\boldsymbol{A}_n$ be n misaligned areas. There are domain transformations that lie in a certain Lie group $\mathbb{G}$: $\mathbb{R}^2 \rightarrow \mathbb{R}^2$, say τ_1, τ_2,..., $\tau_n \in \mathbb{G}$ transform all the misaligned areas to well-aligned $\boldsymbol{A}_1 \circ \tau_1$, $\boldsymbol{A}_2 \circ \tau_2$,..., $\boldsymbol{A}_n \circ \tau_n$. By integrating them in the form of matrix, we have $\boldsymbol{A} \circ \Gamma = [\mathrm{vec}(\boldsymbol{A}_1 \circ \tau_1) | \mathrm{vec}(\boldsymbol{A}_2 \circ \tau_2) | ... | \mathrm{vec}(\boldsymbol{A}_n \circ \tau_n)]$, where Γ contains the n transformations τ_1, τ_2,..., τ_n.

As we know from the work of RASL, enforcing low-rankness among multiple frames will still leave the ambiguity of a common 2D transformation on all the frames. To fix this, we may assume that the region of interest has some spatial structures and at its "normalized" pose, the region as a 2D matrix achieves the lowest-rank.[1] Based on these assumptions, we can naturally formulate our task as the fol-

[1]Please note that, the 2D region does not have to be a strictly low-rank texture for this to be helpful for resolving the ambiguity.

lowing optimization problem:

$$\min_{\boldsymbol{B}} \operatorname{rank}(\boldsymbol{B}) + \omega \sum_{i=1}^{n} \operatorname{rank}(\mathcal{R}(\boldsymbol{B}(i))) + \lambda \|\boldsymbol{E}\|_0, \quad \text{s.t. } \boldsymbol{B} + \boldsymbol{E} = \boldsymbol{A} \circ \Gamma, \tag{1}$$

where $||\cdot||_0$ denotes the ℓ^0-norm, ω and λ are the coefficients controlling the weights of the terms of the rank of each individual area in spatial space and the sparse matrix $\boldsymbol{E}$, and $\mathcal{R}(\cdot)$ represents the linear operator of reshaping a vector back to its original 2D form.

Let us take a closer look at the optimization problem (1). The objective function is non-convex due to the non-convexity of the rank function and the ℓ^0 norm. It is extremely difficult (NP-hard and hard to approximate) to minimize such an objective. Alternatively, minimizing the natural convex surrogate for the objective function in (1) can exactly recover the low rank matrix $\boldsymbol{B}$, as long as the rank of the matrix $\boldsymbol{B}$ to be recovered is not too high and the number of non-zero elements in $\boldsymbol{E}$ is not too large [1]. The convex relaxation replaces $\operatorname{rank}(\cdot)$ and the ℓ^0 norm with the nuclear norm and the ℓ^1 norm, respectively. Another obstacle to solve the problem (1) is the nonlinearity of the constraint $\boldsymbol{B} + \boldsymbol{E} = \boldsymbol{A} \circ \Gamma$, because of the complicated dependence of $\boldsymbol{A} \circ \Gamma$ on Γ. Linearization is a popular technique to approximate the original constraint around the current estimate when the transformations change slightly. The linearized constraint can be written as $\boldsymbol{B} + \boldsymbol{E} = \boldsymbol{A} \circ \Gamma + \sum_{i=1}^{n} \boldsymbol{J}_i \Delta\Gamma \epsilon_i \epsilon_i^T$, where $\boldsymbol{J}_i$ is the Jacobian of the i^{th} area with respect to the transformation parameters τ_i and $\{\epsilon_i\}$ denotes the standard basis for $\mathbb{R}^n$. By putting the convex relaxation and the linearization together, the optimization problem (1) turns out to be like:

$$\min_{\boldsymbol{B}} \|\boldsymbol{B}\|_* + \omega \sum_{i=1}^{n} \|\mathcal{R}(\boldsymbol{B}(i))\|_* + \lambda \|\boldsymbol{E}\|_1, \quad \text{s.t.} \quad \boldsymbol{B} + \boldsymbol{E} = \boldsymbol{A} \circ \Gamma + \sum_{i=1}^{n} \boldsymbol{J}_i \Delta\Gamma \epsilon_i \epsilon_i^T. \tag{2}$$

The optimal solution to (2) can not be obtained directly, since the linearization is only a local approximation to the original. We solve it iteratively to converge to a (local) minimum of the original non-convex problem.

The major task now is to solve the inner loop, *i.e.* the linearized convex optimization problem (2). The Augmented Lagrange Multiplier (ALM) with Alternating Direction Minimizing (ADM) strategy is an efficient and effective solver [10]. But, in our problem (2), there are two terms with respect to $\boldsymbol{B}$. To separate them, we introduce an auxiliary variable $\boldsymbol{S}$ to replace $\boldsymbol{B}$ in the second term of the objective function (2). Accordingly, $\boldsymbol{B} = \boldsymbol{S}$ is as an additional constraint. Thus, the augmented Lagrangian function is given by:

$$\mathcal{L}_\mu(\boldsymbol{B}, \boldsymbol{E}, \boldsymbol{S}, \Delta\Gamma, \boldsymbol{Y}, \boldsymbol{Q}) = \|\boldsymbol{B}\|_* + \omega \sum_{i=1}^{n} \|\mathcal{R}(\boldsymbol{S}(i))\|_* + \lambda \|\boldsymbol{E}\|_1 + \langle \boldsymbol{Y}, \boldsymbol{M} \rangle + \frac{\mu}{2} \|\boldsymbol{M}\|_F^2 + \langle \boldsymbol{Q}, \boldsymbol{O} \rangle + \frac{\mu}{2} \|\boldsymbol{O}\|_F^2,$$

where we define $\boldsymbol{M} = \boldsymbol{A} \circ \Gamma + \sum_{i=1}^{n} \boldsymbol{J}_i \Delta\Gamma \epsilon_i \epsilon_i^T - \boldsymbol{B} - \boldsymbol{E}$ and $\boldsymbol{O} = \boldsymbol{S} - \boldsymbol{B}$, $\boldsymbol{Y}$ and $\boldsymbol{Q}$ are Lagrange multiplier matrices, μ is a positive penalty scalar, $\langle \cdot, \cdot \rangle$ denotes the matrix inner product, and $\|\cdot\|_F$ represents the Frobenius norm. Besides the Lagrange multipliers, there are four variable terms, *i.e.* $\boldsymbol{B}$, $\boldsymbol{E}$, $\boldsymbol{S}$ and $\Delta\Gamma$. It is difficult to simultaneously optimize them. So we approximately solve it in the manner of minimizing one variable at a time, *i.e.* Alternating Direction Minimizing (ADM) strategy, as following:

$$\boldsymbol{B}_{k+1} = \operatorname*{argmin}_{\boldsymbol{B}} \mathcal{L}_{\mu_k}(\boldsymbol{B}, \boldsymbol{E}_k, \boldsymbol{S}_k, \Delta\Gamma_k, \boldsymbol{Y}_k, \boldsymbol{Q}_k), \tag{3}$$

$$\boldsymbol{E}_{k+1} = \operatorname*{argmin}_{\boldsymbol{E}} \mathcal{L}_{\mu_k}(\boldsymbol{B}_{k+1}, \boldsymbol{E}, \boldsymbol{S}_k, \Delta\Gamma_k, \boldsymbol{Y}_k, \boldsymbol{Q}_k), \tag{4}$$

$$\boldsymbol{S}_{k+1} = \operatorname*{argmin}_{\boldsymbol{S}} \mathcal{L}_{\mu_k}(\boldsymbol{B}_{k+1}, \boldsymbol{E}_{k+1}, \boldsymbol{S}, \Delta\Gamma_k, \boldsymbol{Y}_k, \boldsymbol{Q}_k), \tag{5}$$

$$\Delta\Gamma_{k+1} = \operatorname*{argmin}_{\boldsymbol{\Delta\Gamma}} \mathcal{L}_{\mu_k}(\boldsymbol{B}_{k+1}, \boldsymbol{E}_{k+1}, \boldsymbol{S}_{k+1}, \Delta\Gamma, \boldsymbol{Y}_k, \boldsymbol{Q}_k), \tag{6}$$

$$\boldsymbol{Y}_{k+1} = \boldsymbol{Y}_k + \mu_k \boldsymbol{M}_{k+1}, \boldsymbol{Q}_{k+1} = \boldsymbol{Q}_k + \mu_k \boldsymbol{O}_{k+1},$$

where $\{\mu_k\}$ is a monotonically increasing positive sequence. For the subproblems (3)-(6), each has a simple closed-form solution, and hence, can be solved efficiently. For convenience, we denote $\boldsymbol{T} = \boldsymbol{A} \circ \Gamma + \sum_{i=1}^{n} \boldsymbol{J}_i \Delta\Gamma_k + \mu_k^{-1} \boldsymbol{Y}_k$. The solutions are:

$$(\boldsymbol{U}_B, \boldsymbol{\Sigma}_B, \boldsymbol{V}_B) = \operatorname{svd}(\frac{1}{2}(\boldsymbol{T} - \boldsymbol{E}_k + \boldsymbol{S}_k + \frac{\boldsymbol{Q}_k}{\mu_k})),$$

$$\boldsymbol{B}_{k+1} = \boldsymbol{U}_B \mathcal{S}_{\frac{1}{2\mu_k}}(\boldsymbol{\Sigma}_B) \boldsymbol{V}_B^T;$$

$$\boldsymbol{E}_{k+1} = \mathcal{S}_{\frac{\lambda}{\mu_k}}(\boldsymbol{T} - \boldsymbol{B}_{k+1});$$

$$\forall i \in [1...n] : (\boldsymbol{U}_i, \boldsymbol{\Sigma}_i, \boldsymbol{V}_i) = \operatorname{svd}(\mathcal{R}(\boldsymbol{B}_{k+1}(i) - \frac{\boldsymbol{Q}_k(i)}{\mu_k})),$$

$$\boldsymbol{S}_{k+1}(i) = \operatorname{vec}(\boldsymbol{U}_i \mathcal{S}_{\frac{\omega}{\mu_k}}(\boldsymbol{\Sigma}_i) \boldsymbol{V}_i^T);$$

$$\Delta\Gamma_{k+1} = \sum_{i=1}^{n} \boldsymbol{J}_i^\dagger (\boldsymbol{B}_{k+1} + \boldsymbol{E}_{k+1} - \boldsymbol{A} \circ \Gamma - \mu_k^{-1} \boldsymbol{Y}_k) \epsilon_i \epsilon_i^T;$$

where $\mathcal{S}_\varepsilon(\cdot)$ is the shrinkage operator: $\mathcal{S}_\varepsilon(x) = \operatorname{sgn}(x) \max(|x| - \varepsilon, 0)$. The extension of the shrinkage operator to vectors and matrices is applying it element-wise. $\boldsymbol{J}^\dagger$ denotes the Moore-Penrose pseudoinverse of $\boldsymbol{J}$.

The entire algorithm of solving the problem (2) has been summarized as Algorithm 1. The outer loop of Algorithm 1 terminates when the change of objective function value between two neighboring iterations is sufficiently

Algorithm 1: Batch Frame Alignment

Input: $\lambda > 0, \omega > 0$, n misaligned and non-rectified regions $\boldsymbol{A} = [\mathrm{vec}(\boldsymbol{A}_1)|\,\mathrm{vec}(\boldsymbol{A}_2)|...|\,\mathrm{vec}(\boldsymbol{A}_n)]$ and initial transformations $\Gamma = [\tau_1|\tau_2|...|\tau_n]$ obtained by feature matching and RANSAC.

while *not converged* **do**

$k = 0; \boldsymbol{Y}_0 = \boldsymbol{B}_0 = \boldsymbol{E}_0 = \boldsymbol{0}; \Delta\Gamma_k = \boldsymbol{0}$;

$\mu_0 > 0, \rho > 1$;

Compute the warped areas $\boldsymbol{A} \circ \Gamma$ and their Jacobians $\boldsymbol{J}_i = \frac{\partial}{\partial \tau_i}\mathrm{vec}(\boldsymbol{A}_i \circ \tau_i)$;

while *not converged* **do**

$\boldsymbol{T} = \boldsymbol{A} \circ \Gamma + \sum_{i=1}^{n} \boldsymbol{J}_i \Delta\Gamma_k + \frac{\boldsymbol{Y}_k}{\mu_k}$;

$(\boldsymbol{U}_B, \boldsymbol{\Sigma}_B, \boldsymbol{V}_B) = \mathrm{svd}(\frac{1}{2}(\boldsymbol{T} - \boldsymbol{E}_k + \boldsymbol{S}_k + \frac{\boldsymbol{Q}_k}{\mu_k}))$;

$\boldsymbol{B}_{k+1} = \boldsymbol{U}_B \mathcal{S}_{\frac{1}{2\mu_k}}(\boldsymbol{\Sigma}_B)\boldsymbol{V}_B^T$;

$\boldsymbol{E}_{k+1} = \mathcal{S}_{\frac{\lambda}{\mu_k}}(\boldsymbol{T} - \boldsymbol{B}_{k+1})$;

for $i = 1 : n$ **do**

$(\boldsymbol{U}_i, \boldsymbol{\Sigma}_i, \boldsymbol{V}_i) = \mathrm{svd}(\mathcal{R}(\boldsymbol{B}_{k+1}(i) - \frac{\boldsymbol{Q}_k(i)}{\mu_k}))$;

$\boldsymbol{S}_{k+1}(i) = \mathrm{vec}(\boldsymbol{U}_i \mathcal{S}_{\frac{\omega}{\mu_k}}(\boldsymbol{\Sigma}_i)\boldsymbol{V}_i^T)$;

end

$\boldsymbol{H} = \boldsymbol{B}_{k+1} + \boldsymbol{E}_{k+1} - \boldsymbol{A} \circ \Gamma$;

$\Delta\Gamma_{k+1} = \sum_{i=1}^{n} \boldsymbol{J}_i^{\dagger}(\boldsymbol{H} - \mu_k^{-1}\boldsymbol{Y}_k)\epsilon_i\epsilon_i^T$;

$\boldsymbol{Y}_{k+1} = \boldsymbol{Y}_k + \mu_k(\sum_{i=1}^{n} \boldsymbol{J}_i \Delta\Gamma_{k+1}\epsilon_i\epsilon_i^T - \boldsymbol{H})$;

$\boldsymbol{Q}_{k+1} = \boldsymbol{Q}_k + \mu_k(\boldsymbol{S}_{k+1} - \boldsymbol{B}_{k+1})$;

$\mu_{k+1} = \rho\mu_k; k = k + 1$;

end

$\Gamma = \Gamma + \Delta\Gamma_k$;

end

Output: Optimal solution ($\boldsymbol{B}^* = \boldsymbol{B}_k$)

small (*e.g.* 0.01) or the maximal number of outer iterations is reached. The inner loop is stopped when $\|\boldsymbol{A} \circ \Gamma + \sum_{i=1}^{n} \boldsymbol{J}_i \Delta\Gamma_k \epsilon_i \epsilon_i^T - \boldsymbol{B}_k - \boldsymbol{E}_k\|_2 \leq \delta\|\boldsymbol{A} \circ \Gamma + \sum_{i=1}^{n} \boldsymbol{J}_i \Delta\Gamma_k \epsilon_i \epsilon_i^T\|$ with $\delta = 10^{-7}$ or the maximal number of inner iterations is reached.

2.2. New Single Frame Alignment

As the above optimization involves expensive matrix optimization, it is not desirable to run it on a whole long sequence. Given a set of n frames are already aligned as above, when a new frame, say $\boldsymbol{a}$ arrives, we can try to rectify and align it to the basis $\boldsymbol{B}$ obtained from Algorithm 1, subject to some error $\boldsymbol{e}$. In general, the well-aligned and rectified region can be represented by a linear combination of the columns from the basis, *i.e.* $\boldsymbol{Bx}$, where $\boldsymbol{x}$ is a coefficient vector. So the formulation of aligning a new frame is:

$$\min \|\boldsymbol{e}\|_0 \quad \mathrm{s.t.} \quad \boldsymbol{Bx} + \boldsymbol{e} = \boldsymbol{a} \circ \tau, \tag{7}$$

Algorithm 2: Single Frame Alignment

Input: a basis $\boldsymbol{B}$ from initial batch, a misaligned and non-rectified region $\boldsymbol{a}$ in a new frame, initial transformation matrix τ (obtained by feature matching and RANSAC) and $\rho > 1$.

Initialization: initial support $\Omega = \boldsymbol{1}^m$;

while *not converged* **do**

$k = 0; \boldsymbol{y}_0 = \boldsymbol{0}; \boldsymbol{x}_0 = \boldsymbol{0}; \boldsymbol{e}_0 = \boldsymbol{0}; \Delta\tau_0 = \boldsymbol{0}; \mu_0 > 0$;

Compute the warped area $\boldsymbol{a} \circ \tau$ and its Jacobian $\boldsymbol{J} = \frac{\partial}{\partial \tau}\mathrm{vec}(\boldsymbol{a} \circ \tau)$;

while *not converged* **do**

$\boldsymbol{t} = \boldsymbol{a} \circ \tau + \boldsymbol{J}\Delta\tau_k + \mu_k^{-1}\boldsymbol{y}_k$;

$\boldsymbol{x}_{k+1} = P_\Omega(\boldsymbol{B})^{\dagger} P_\Omega(\boldsymbol{t} - \boldsymbol{e}_k)$;

$\boldsymbol{e}_{k+1} = \mathcal{S}_{\mu_k^{-1}}(P_\Omega(\boldsymbol{t} - \boldsymbol{B}\boldsymbol{x}_{k+1}))$;

$\boldsymbol{h} = \boldsymbol{B}\boldsymbol{x}_{k+1} + \boldsymbol{e}_{k+1} - \boldsymbol{a} \circ \tau$;

$\Delta\tau_{k+1} = P_\Omega(\boldsymbol{J})^{\dagger} P_\Omega(\boldsymbol{h} - \mu_k^{-1}\boldsymbol{y}_k)$;

$\boldsymbol{y}_{k+1} = \boldsymbol{y}_k + \mu_k P_\Omega(\boldsymbol{J}\Delta\tau_{k+1} - \boldsymbol{h})$;

$\mu_{k+1} = \rho\mu_k; k = k + 1$;

end

$\tau = \tau + \Delta\tau_k; \Omega = \mathrm{support}(\boldsymbol{a} \circ \tau - \boldsymbol{B}\boldsymbol{x}_k)$;

end

Output: Optimal solution ($\boldsymbol{x}^* = \boldsymbol{x}_k; \tau^* = \tau$)

where $\boldsymbol{e}$ represents the residual, and τ is the transformation matrix of $\boldsymbol{a}$. The formulation of (7) has been noticed earlier in the field of face recognition [17][18]. But they do not explicitly give the solution in a manner of ALM. In this section, we intend to detail an algorithm for solving the problem based on ALM and ADM.

To make the problem (7) tractable, similarly with (1), we again employ $\|\boldsymbol{e}\|_1$ as the surrogate of $\|\boldsymbol{e}\|_0$, and linearize the constraint. The ALM method gives us:

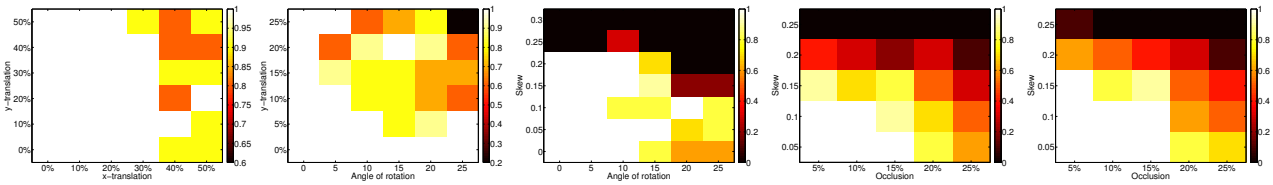
$$\mathcal{L}_\mu(\boldsymbol{x}, \boldsymbol{e}, \Delta\tau, \boldsymbol{y}) = \|\boldsymbol{e}\|_1 + \langle \boldsymbol{y}, \boldsymbol{a} \circ \tau + \boldsymbol{J}\Delta\tau - \boldsymbol{B}\boldsymbol{x} - \boldsymbol{e}\rangle + \frac{\mu}{2}\|\boldsymbol{a} \circ \tau + \boldsymbol{J}\Delta\tau - \boldsymbol{B}\boldsymbol{x} - \boldsymbol{e}\|_2^2.$$

With the help of the ADM strategy, we have the iteration:

$$\begin{aligned}
\boldsymbol{x}_{k+1} &= \boldsymbol{B}^{\dagger}(\boldsymbol{a} \circ \tau + \boldsymbol{J}\Delta\tau_k + \mu_k^{-1}\boldsymbol{y}_k - \boldsymbol{e}_k),\\
\boldsymbol{e}_{k+1} &= \mathcal{S}_{\mu_k^{-1}}[\boldsymbol{a} \circ \tau + \boldsymbol{J}\Delta\tau_k + \mu_k^{-1}\boldsymbol{y}_k - \boldsymbol{B}\boldsymbol{x}_{k+1}],\\
\Delta\tau_{k+1} &= \boldsymbol{J}^{\dagger}(\boldsymbol{B}\boldsymbol{x}_{k+1} + \boldsymbol{e}_{k+1} - \boldsymbol{a} \circ \tau - \mu^{-1}\boldsymbol{y}_k),\\
\boldsymbol{y}_{k+1} &= \boldsymbol{y}_k + \mu_k(\boldsymbol{a} \circ \tau + \boldsymbol{J}\Delta\tau_{k+1} - \boldsymbol{B}\boldsymbol{x}_{k+1} - \boldsymbol{e}_{k+1}).
\end{aligned}$$

When the region to be aligned is corrupted by sparse occlusions or corruptions, the procedure proposed above can handle well. If the corruption is out of the range that the robust linear regression can make up, the recovered regressor $\boldsymbol{x}$ may not be so reasonable. In such case, we gradually remove some of the pixels that have very large reconstruction errors, and run the above robust alignment again on the remaining region. The associated problem (7) becomes:

$$\min_{\boldsymbol{x}} \|P_\Omega(\boldsymbol{e})\|_0, \quad \mathrm{s.t.} \quad P_\Omega(\boldsymbol{Bx} + \boldsymbol{e}) = P_\Omega(\boldsymbol{a} \circ \tau), \tag{8}$$



(a) (b) (c) (d) (e)

Figure 4. Quantitative performance analysis of Algorithm 1 on simulated data. (a) x-translation vs. y-translation. (b) angle of rotation vs. y-translation. (c) angle of rotation vs. skew. (d) occlusion rate vs. skew. (e) occlusion rate vs. skew using more images. Lighter regions indicate higher probabilities of success, while darker ones stand for lower.

where $\Omega \in \{0,1\}^m$ is the support for the good pixels, $P_\Omega(\cdot)$ is the operator that keeps the elements of vector (the rows of matrix) with $\Omega_i = 1$ and turns those with $\Omega_i = 0$ into zeros.

If Ω and τ are provided exactly, then $\boldsymbol{x}$ can be estimated by linear regression. However, τ is also unknown, and the intermediate estimation of the support Ω could be wrong in a subset of its entries. Fortunately, if the initialization of τ and the guess of Ω are reasonable, the algorithm converges reliably to the optimal alignment. In our experiments, τ is initialized by feature matching and RANSAC [5]. Here, we give an efficient and effective support estimation method. To estimate Ω, we assume the distribution of the values in $\boldsymbol{e}$ satisfies the Gaussian distribution (η, σ). Note that, the area does not contain any occlusions or corruptions if the values of the elements in $\boldsymbol{e}$ are all very close to zero. Therefore, we have $t = \max(\eta + \sigma, \beta)$ (in our experiences, $\beta = 0.1$ works practically well). Then the update of Ω is done via: if $|\boldsymbol{e}_i| < t$, $\Omega_i = 1$; $\Omega_i = 0$ otherwise. The procedure of aligning a new frame is summarized in Algorithm 2. The stopping criteria of Algorithm 2 are in the same spirit as Algorithm 1.

2.3. Foreground Separation

So far, we have the recovered temporally aligned and spatially rectified area and the associated error residual. If our goal is only to recover a clean texture of the region, remove the occluding foreground, and replace the region with another texture throughout the sequence, then results from the above two steps would be sufficient. However, if we want to super-impose the occluding foreground back to the edited region to achieve more realistic visual effect, we have to precisely segment out the foreground. For images, one pixel presents either background or foreground[2]. Notice that the error residual $\boldsymbol{e}$ that we have obtained from the above robust alignment is not the foreground per se: it is the difference between the original images and the background. Nevertheless, the error residual is very informative about where the occluding foreground is: in general, we get large residual errors around occluding pixels; and small ones around background pixels. Therefore, we can initially assign a pixel to foreground if the error is above certain threshold and to background if the error is small enough. For all pixels whose errors fall between the two thresholds, we label them as unknown. In this way, we obtain an initial trimap for foreground segmentation. We then employ existing alpha matting methods [9][2][15] to obtain a more precise segmentation of the foreground from the image frames.

[2]In this paper, we discuss the cases that the objects of both background and foreground are opaque.

3. Simulations and Experiments

In this section, we demonstrate the efficacy of our method on both simulated and real data. The free parameters in our framework consist of ω and λ in Eq. (2). Unless otherwise stated, the two parameters are fixed through our experiments: $\omega = 5/n$ and $\lambda = 3/\sqrt{m}$, where n is the number of the images used to construct the area basis and m is the amount of pixels in the area of interest.

We first verify the ability of the batch frame alignment Algorithm 1 to cope with varying levels of corruptions on a (rectified) checker-board pattern with white, gray and black blocks. Translation, rotation, skew, occlusion rate and the number of images are the factors to be tested. The task is to align and rectify the images to a 100×100 pixel canonical window. We synthetically perturb each of the input images by affine transformations with skew level s_0, whose angles of rotation are uniformly distributed in the range $[-\theta_0/2, \theta_0/2]$, x- and y- translations are uniformly distributed in the range $[-x_0/2, x_0/2] \times 100$ and $[-y_0/2, y_0/2] \times 100$ pixels respectively. Due to the complex dependence of the algorithm on these factors, we test them separately.

A case is considered successful if the average intersection rate between each of the transformed images and the canonical is greater than 96%. Figure 4 shows the successful probability map over 10 independent trials using 25 images. Figure 4 (a) is with $\theta_0 = 0$ and $s_0 = 0$ fixed, and varying levels of translation x_0 and y_0. Our algorithm always correctly finish the task when x-translation and y-translation up to 20% and 50% respectively. Even

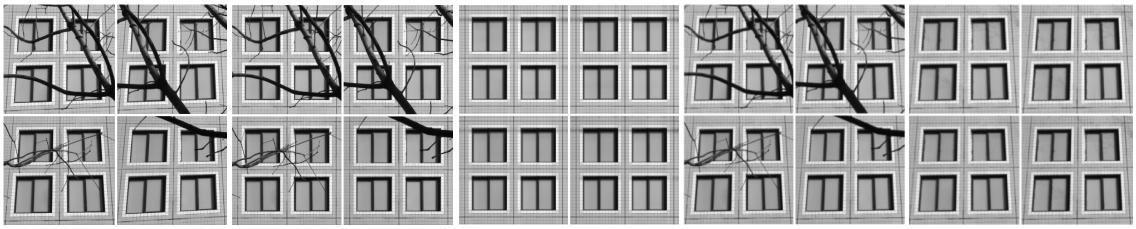


Figure 5. Result comparison between RASL and our method on a real-world window case.

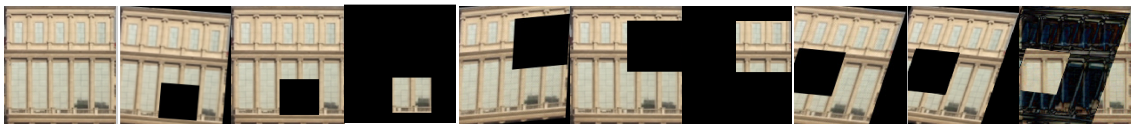


Figure 6. Qualitative results of the proposed single frame alignment method (Algorithm 2) with respect to translation, rotation, skew and occlusion. The first picture is the reference. Every three pictures of the rest are in one group, which represent the deformed and polluted version of the reference, the transformed result by our proposed and the corresponding residual, respectively.

x-translation grows to 30%, the successful rates are reasonable with y-translations under 50%. Similarly, in Figure 4 (b), the results are obtained by fixing $s_0 = 0$ and $x_0 = 10\%$, and varying y_0 and θ_0. As can be seen in the figure, if y_0 decreases to 0%, the algorithm can achieve the goal even when the angle of rotation is up to 25°. As both of y_0 and θ_0 increase, the probability of success decreases. To take into account the skew factor, we constantly set both x_0 and y_0 to be 5%, and test θ_0 and s_0 with different values. The results are shown in Fig. 4 (c). Notice that the algorithm can tolerate up to 10° rotation and 0.2 skew level, expect for the one of 10° rotation and 0.15 skew, which is with the successful rate 80%.

We repeat the above procedure to further verify the performance of the proposed algorithm with respect to different levels of occlusion and the number of images. Figure 4 (d) shows the robustness of the proposed algorithm to random occlusion and skew level with $x_0 = 5\%$, $y_0 = 10\%$ and $\theta_0 = 10^\circ$. It indicates that our algorithm can converge well up to 15% random occlusion and 0.1 skew. Although the case of 0.1 skew and 15% occlusion does not succeed in every trial, it has 90% successful rate. Compared to Fig. 4 (d) that uses 25 images, the performance improves in Fig. 4 (e) where 35 images are used. This is because with more images, the redundancy in the data is higher and hence, the low-rank model fits better.

The comparison to RASL is shown in Fig. 5. The first column shows the sample images of a window with various transformations and occlusions. The second is the result of aligned and rectified windows without removing occlusions by our method. The third is the recovered clean regions $\boldsymbol{B}$ by our method. The last two images are the results obtained by the RASL code obtained from the authors [12]. Another example is shown in Fig. 3 (a)-(c), which are the medians of the input areas, clean versions processed by RASL and our method, respectively. Furthermore, Figure 3 (d)-(e) simply demonstrate the advantage of the rectification operation from the perspective of virtual reality.

Note that due to the limit of space, we do not quantitatively analyze the performance of the single frame alignment Algorithm 2 here as it is designed in a similar spirit as Algorithm 1. The algorithm performs rather well in practice, with respect to the transformation and the occlusion. Figure 6 gives some qualitative results of the algorithm. We synthesize deformed and polluted images by controlling parameters (x-translation, y-translation, angle of rotation, skew, occlusion fraction) based on the reference, *i.e.* the first facade shown in Fig. 6. The three groups respectively correspond to the parameters $(10\%, 10\%, 10^\circ, 0, 15\%)$, $(10\%, 10\%, 10^\circ, 0.1, 25\%)$ and $(15\%, 15\%, 15^\circ, 0.15, 20\%)$. From the results, we see that the first two cases succeeded, while the last one failed due to the simultaneously very large deformation and intensive occlusion.

Figure 7 demonstrates improved robust alignment and recovery results with occlusion detection (as discussed at the end of Section 2.2). The top row is the recovery result by using all the pixels in the region, and the middle row is the result with occlusion detection and masking. Another merit of our method is that it preserves all global illumination changes in the original frames, which can be seen from the comparison with the median facade shown in the middle of the bottom row. This property ensures that visual realism of the original sequence can be maximally preserved. If we directly use the area obtained by feature matching and RANSAC estimation, the result will be less accurate and unreal, like the example shown in the bottom row of Fig. 7.

Figure 8 shows an example of foreground separation

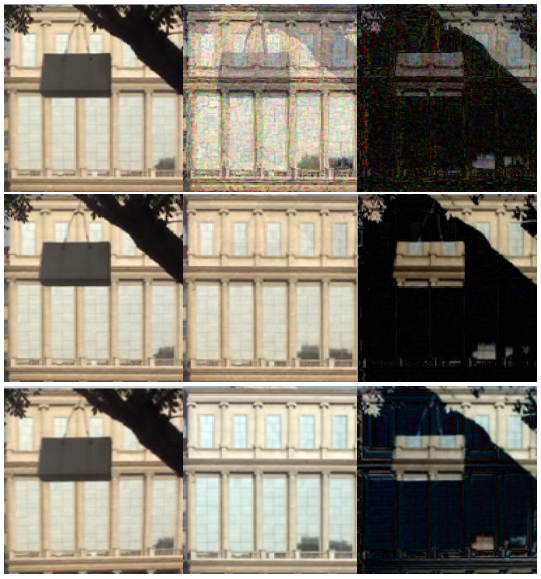


Figure 7. Comparison of robust alignment results on an example with deformation and large occlusion. **Top and Middle rows:** the recovered results without and with occlusion detection. **Bottom row:** the left is the result by feature matching and RANSAC, the middle is the median area obtained from the area basis, and the right is their residual.

from the recovered background and error residuals (as in Section 2.3). It contains four images (from left to right: the recovered residual, the trimap, the foreground mask, and the foreground cutout respectively). From the residual, we can determine which pixels are foreground. With the help of texture cue, we can determine that the pixels in the original image having similar textures with their corresponding pixels in the recovered image are background. The rest is classified as unknown. Hence, the trimap is automatically constructed and used as the input to alpha matting. In this example, we adopt dense SIFT to measure the texture similarity between the original image and the recovered. The alpha matting result is computed by using the technique introduced in [9]. Readers can adopt other cues to determine the background and unknown, and choose other texture similarity measurements and alpha matting methods for the task of foreground separation.

Finally, we apply our framework on several real-world image sequences as shown in Fig. 9. In each of the three cases, the first row displays sample images from the same sequences, and the second gives the edited results by the proposed method. The top case changes the building facade, the middle one changes the monitor background of the laptop, and the bottom one repairs the building facade texture as well as adding a new advertisement banner. In all the three cases, the results preserve the photo realism of the original sequence, including global lighting and reflection on the monitor etc. The virtual reality achieved by our method is also rather striking, not only the pose and geometry of the edited area are consistent throughout the sequence, but also all the occlusions in the original sequence are correctly synthesized on the newly edited regions (*e.g.* Top: the tree trunk and branches; Middle: the orange, the toy, the box and the screen reflection; Bottom: the traffic light poles). Using our system, the only human intervention needed to achieve these tasks is to specify the edited area in the first frame and provide the replacement texture.

Figure 8. Illustration of automatic foreground separation.

4. Conclusion

We have presented a new framework that simultaneously align and rectify image regions in a video sequence, as well as automatically construct trimaps and segment foregrounds. Our framework leverages the recent advances in robust recovery of a high-dimensional low-rank matrix despite gross sparse errors. The system can significantly reduce the interaction from users for editing certain areas in the video. The quantitative analysis revealed that our method can be applied under a wide range of conditions. The experiments on real world sequences also demonstrated the robust performance of the proposed framework.

Acknowledgements

This work was supported by National High-tech R&D Program of China(2012AA011503), National Natural Science Foundation of China (No. 61003200), and Tianjin Key Technologies R&D program (No.11ZCKFGX00800).

References

[1] E. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM*, 58(3):1–37, 2011.

[2] Q. Chen, D. Li, and C. Tang. KNN matting. In *Proc. of CVPR*, pages 869–876, 2012.

[3] M. Cox, S. Lucey, S. Sridharan, and J. Cohn. Least squares congealing for unsupervised alignment of images. In *Proc. of CVPR*, pages 1–8, 2008.

[4] M. Cox, S. Lucey, S. Sridharan, and J. Cohn. Least-squares congealing for large numbers of images. In *Proc. of ICCV*, pages 1–8, 2009.

[5] M. Fischler and R. Bolles. Random sample consensus: A paradigm for model tting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

Figure 9. Three cases of editing on image sequences. **Top:** Outdoor scene with the building facade changing. **Middle**: Indoor scene with the laptop's monitor background changing. **Bottom:** Outdoor scene with repairing and advertising. More details can be found in the text.

[6] X. Hou and L. Zhang. Color conceptualization. In *Proc. of ACM Multimedia*, pages 265–268, 2007.

[7] G. Huang, V. Jain, and E. Learned-Miller. Unsupervised joint alignment of complex images. In *Proc. of ICCV*, pages 1–8, 2007.

[8] E. Learned-Miller. Data driven image models through continuous joint alignment. *IEEE TPAMI*, 28(2):236–250, 2006.

[9] A. Levin, D. Lischinski, and Y. Weiss. A closed-form solution to natural image matting. *IEEE TPAMI*, 30(2):1–15, 2008.

[10] Z. Lin, M. Chen, L. Wu, and Y. Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. Technical Report UILU-ENG-09-2215, UIUC Technical Report, 2009.

[11] A. Nigeau, M. Bertalmio, V. caselles, and G. Sapiro. A comprehensive framework for image inpainting. *IEEE Trans. on Image Processing*, 19(10):2634–2645, 2010.

[12] Y. Peng, A. Ganesh, J. Wright, W. Xu, and Y. Ma. RASL: Robust alignment by sparse and low-rank decomposition for linearly correlated images. *IEEE TPAMI*, 34(11):2233–2246, 2012.

[13] P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. In *ACM SIGGRAPH*, pages 313–318, 2003.

[14] Y. Pritch, E. Kav-Venaki, and S. Peleg. Shift-map image editing. In *Proc. of ICCV*, pages 151–158, 2009.

[15] J. Sun, J. Jia, C. Tang, and H. Shum. Poisson matting. In *ACM SIGGRAPH*, pages 315–321, 2004.

[16] A. Vedaldi, G. Guidi, and S. Soatto. Joint alignment up to (lossy) transforamtions. In *Proc. of CVPR*, pages 1–8, 2008.

[17] A. Wagner, J. Wright, A. Ganesh, Z. Zhou, H. Mobahi, and Y. Ma. Toward a practical face recognition system: Robust alignment and illumination by sparse representation. *IEEE TPAMI*, 34(2):373–386, 2012.

[18] A. Yang, A. Ganesh, Z. Zhou, S. Sastry, and Y. Ma. A review of fast l1-minimization algorithms for robust face recognition. *CoRR abs/1007.3753*, 2010.

[19] Z. Zhang, A. Ganesh, X. Liang, and Y. Ma. TILT: Transform-invariant low-rank textures. *IJCV*, 99(1):1–24, 2012.