IPSJ Transactions on Computer Vision and Applications Vol.4 170–178 (Dec. 2012)

[DOI: 10.2197/ipsjtcva.4.170]

Research Paper

Real-time Video Mosaicing using Non-rigid Registration

Rafael Henrique Castanheira de Souza[1,a) Masatoshi Okutomi[1,b)
Akihiko Torii[1,c)

Received: April 11, 2012, Accepted: October 1, 2012, Released: December 25, 2012

Abstract: This paper presents a real-time incremental mosaicing method that generates a large seamless 2D image by stitching video key-frames as soon as they are detected. There are four main contributions: (1) we propose a "fast" key-frame selection procedure based solely on the distribution of the distance of matched feature descriptors. This procedure automatically selects key-frames that are used to expand the mosaics while achieving real-time performance; (2) we register key-frame images by using a non-rigid deformation model based on a triangular mesh in order to "smoothly" stitch images when scene transformations can not be expressed by homography; (3) we add a new constraint on the non-rigid deformation model that penalizes over-deformation in order to create mosaics with natural appearance; (4) we propose a fast image stitching algorithm for real-time mosaic rendering modeled as an instance of the minimum graph cut problem, applied to mesh triangles instead of the image pixels. The performance of the proposed method is validated by experiments in non-controlled conditions and by comparison with a state-of-the-art method.

Keywords: non-rigid registration, image stitching, graph cut

1. Introduction

Mosaicing is a classical application of image registration. Typically, a set of images is stitched together to simulate a camera with a larger field of view. Real-time mosaicing can be useful for medical imaging, augmented reality, digital camera panorama generation, etc. [12].

Classical mosaicing methods work under the assumption that the input images are related to each other by homography (*projective transformation*). This assumption holds true when the images are acquired under some limited conditions (camera rotation around its optical center or scene lying on a planar surface). Unless these conditions are satisfied, the images can not be perfectly aligned by registration and the results may be very poor. This problem may be alleviated by the application of non-rigid registration [10]. In this work, we propose a method of online mosaicing that can generate 2D mosaics from video inputs acquired beyond homography assumptions.

A naive approach to online mosaicing is to register and stitch the current video key-frame into the previously selected key-frame. The process will accumulate registration error which will grow with each new image added to the sequence causing the final mosaic to look over-deformed. Also, when using non-rigid registration, not all regions of the image being registered will have the same alignment precision. It is necessary to apply some robust method of image stitching capable of creating the final mosaic using only the well aligned parts of the registered image while ignoring the misaligned regions. This method must also run in real-time.

This paper presents a method which uses a very efficient feature based non-rigid registration model in order to align images with high precision. At the same time, the over-deformation of the mosaic is avoided during the online mosaic creation. These two objectives are achieved by formulating the registration problem enforcing smoothness while keeping the original proportions of the captured key-frame. Additionally, in order to achieve real-time processing, the key-frames are efficiently extracted from the video by a procedure which uses the distance distribution of matched feature descriptors. This paper also presents a fast image stitching method tailored for feature-based image registration. After the images are registered, they must be stitched together to create a mosaic. Pixel selection is one of the main steps in image stitching. Given a sequence where the key-frames may have many regions of overlap, image stitching consists of deciding which pixels will be used to compose the final mosaic. A common method of pixel selection is the graph cut algorithm [16]. Generally, image stitching with graph cut uses a formulation in which the vertices represent pixels of the images being stitched together. However, this approach is infeasible for a real-time method working with high-resolution images because of the great number of vertices in the resulting graph cut model. In the proposed stitching method, vertices represent triangles of the mesh model used to represent the non-rigid transformations. By these means, since the number of triangles in the mesh model is much smaller than the number of pixels, real-time processing can be achieved in spite of the complex image stitching algorithm.

The paper is organized as follows. In Section 2, related methods are presented. Section 3 presents the proposed method. Section 4 shows the result of the experimental validations. Finally,

[1] Tokyo Institute of Technology, Meguro, Tokyo 152–8550, Japan
[a) rafaelh.souza@ok.ctrl.titech.ac.jp
[b) mxo@ctrl.titech.ac.jp
[c) torii@ctrl.titech.ac.jp

Section 5 presents the conclusions of this work and future research subjects.

2. Related Work

For the reader who is not familiar with mosaicing, Szeliski [12] presents a comprehensive tutorial about a variety of methods of registration and mosaic composition.

Since image mosaicing is a well studied area of computer vision, there are many approaches to this problem. They can be grouped in 3 classes: offline methods that use homography or lower degree transformations, offline methods that use higher degree transformations, and online methods. The first group includes the works Refs. [1], [3], [5], [9], which are based on global transformations such as homography. The second group includes the works Refs. [4], [6], which model the deformation as quadratic functions. The third group, which is the most related to the proposed method, includes the works of Refs. [7], [11]. The work in Ref. [7] uses 3D information for registering aerial images using a non real-time algorithm. The method in Ref. [11] is online and avoids the problem of over-deformation by using fixed camera movements (translation, forward motion, etc.).

Although most of the works dealing with mosaicing make use of global transformations such as homography, there are more general registration methods that use non-rigid deformation. Some of them use feature based methods, e.g., Refs. [2], [8], [10]. Feature based methods are generally more computationally efficient than area based methods [12], specially in the case of non-rigid registration. The method in Ref. [8] can register images in real-time even in the presence of a large ratio of outliers. However, this method is designed for pairs of images only.

Therefore, on top of the state of the art, the contributions of the proposed work are: real-time performance, use of non-rigid registration, prevention of over-deformation of the mosaic, less restrictions on camera movement, and a fast implementation of graph cut for image stitching.

3. Proposed Method

The mosaicing procedure consists of four steps: key-frame selection, feature matching, registration, and mosaic displaying. The key-frame selection module reads the input video and selects which key-frames will be used to create the mosaic. The feature matching module matches the feature points in the last and the previous selected key-frames. The pairwise registration module receives the set of matched features and registers the newly selected key-frame into the previously selected key-frame. The registered key-frame is then sent to the mosaic creation module where it is stitched to the mosaic and displayed. The procedure is repeated again, until the end of the video is reached. The modules are explained in more details in the following sections.

3.1 Key-frame Selection

In order to create mosaics efficiently, only a small subset of the video frames must be selected as key-frames. This key-frame set must be as sparse as possible, to reduce the number of registrations performed. However, at the same time, it must contain enough overlapping key-frames so that a complete mosaic can be composed. To find key-frames with these characteristics, it is necessary to be able to estimate the overlap between two frames. The following procedure can perform this estimation: (1) the features in both frames being compared are detected using SURF descriptors [14]); (2) the nearest-neighbor matching of the features is computed (for efficiency reasons, no outlier rejection is done during this step); (3) a histogram of the distance between the matched descriptors is computed; (4) the overlap measure (OM) is computed.

The OM is a function that estimates the overlap between two images. It is defined as follows:

$$OM(H) = \sum_{j=1}^{n_{Bin}} G((j-0.5)h_{size}, \varsigma)H_j, \qquad (1)$$

where H is the descriptor distance histogram, n_{Bin} is the number of bins in H, h_{size} is the size of each bin, $(j-0.5)h_{size}$ is the average range of the bin j, G is a Gaussian weighting function with 0 mean and standard deviation ς. This weighting function assigns larger weights to distances near zero, and the weight decays quickly, so that the bins which probably contain correct matches receive a larger weight than the bins with wrong matches.

Making use of OM, the key-frames are selected by the following algorithm: (1) the first video frame is selected and used as reference; (2) if the next frame has an OM (regarding the reference frame) which is smaller than a given threshold, it is selected and becomes the new reference. Step (2) is repeated until the end of the video.

It was experimentally observed that the probability distribution of the descriptor distances changes according to the intersection size between the image pair. **Figure 1** (a) shows two frames with a small overlap. The descriptor distance has unimodal distribution (Fig. 1 (c), red histogram). Figure 1 (b) shows two frames with a larger overlap. In this case, the left tail of the distribution increases (Fig. 1 (c), blue histogram). This happens due to

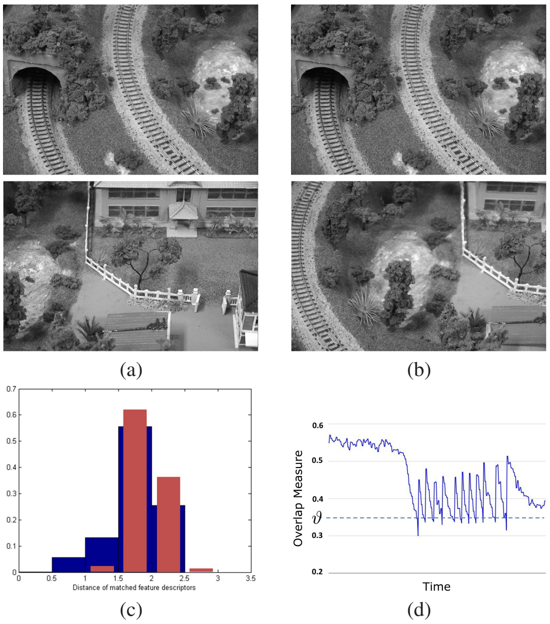


Fig. 1 Key-frame selection. (a) Pair of frames with a small overlap. (b) Pair of frames with a large overlap. (c) Histogram of the distance of matched descriptors: the red bars represent pair (a) and the blue bars the pair (b). (d) Variation of the overlap measure over time.

the greater proportion of inliers among the matched features. Figure 1 shows the variation of OM over time, in a video recorded by a translating camera. The value of OM decreases as the intersection becomes smaller and rises again when a new key-frame is selected. In this algorithm, the parameter ϑ is a threshold that, in the current implementation, is selected manually (see Section 4.1).

3.2 Feature Matching

The feature matches found during key-frame selection can not be used as-is since most of the matches are outliers. A pruning of these outliers is necessary before the matches can be used by the registration procedure. Let M be the set of matches of a consecutive pair key-frames. The set M is initialized by a simple method (presented in Ref. [17]), which is as follows. The set M is initially empty. Each feature in one key-frame is paired with the two most similar features of the other key-frame (the features were already computed during key-frame selection). The descriptor distances of these two matches are calculated. Features whose variation in the distances is large (the smaller distance is less than 60% of the larger distance) are considered inliers and the closest match is added to M.

The idea behind this method is that inlier matches generally present distances much smaller than false matches. Despite of being simple, this pruning method is very efficient, producing a set M with few false positives. However, this method may consider many correct matches as outliers (false negatives). In order to increase the number of matches, epipolar geometry can be used to do a guided matching [15]. The fundamental matrix $\mathcal{F}$ is estimated using M by the 8-point algorithm, using the previously selected inlier matches as initialization. After $\mathcal{F}$ is estimated, the remaining pairs of matched features $c = c_0, c_1 \notin M$ which satisfy $c_0^{\mathrm{T}}\mathcal{F}c_1 < \varepsilon_{Match}$, where ε_{Match} is a small value, are considered inliers and added to M.

3.3 Registration

This section explains the registration model used in the proposed method. Two constraints must be met: the mosaic must be as seamless as possible and over-deformation must be avoided. For doing so, the proposed method applies a non-rigid deformation model that uses triangle meshes and a registration algorithm that uses the feature matchings previously computed.

3.3.1 Deformation Model for Image Registration

A 2D mesh model is used to implement the non-rigid transformations. Each vertex (or control point) v_j is represented by its coordinates (x_j, y_j). The entire mesh is written as $S = [X, Y]^{\mathrm{T}}$, where X is a vector containing the x coordinates of the control points and Y the vector containing the y coordinates. The warp of any point p, which is inside a mesh triangle defined by the vertices v_i, v_j, and v_k, can be calculated using the barycentric coordinates of p: $w(p, S) = \sum_{l\in\{i,j,k\}} B(p, v_l)[x_l, y_l]^{\mathrm{T}}$, where $B(p, v_l)$ is the barycentric coordinate of p in relation to $v_l \in \{v_i, v_j, v_k\}$ (computed in relation to the identity mesh S_0). **Figure 2** illustrates the basic principle of this kind of transformation.

3.3.2 Problem Formulation

The initial model of pairwise non-rigid registration was drawn from Zhu et al.'s work [8], which was based on Pilet et al.'s work [2]. It is summarized by the equation below:

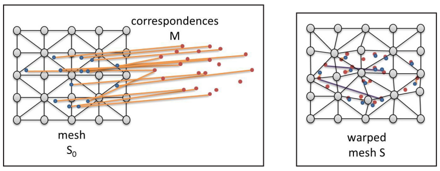


Fig. 2 Deformation using a mesh model. (a) Identity mesh S_0. (b) Mesh S warped to reduce the projection error of the matched features.

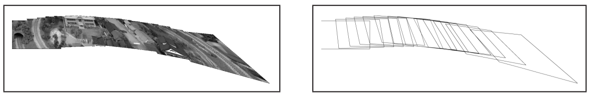


Fig. 3 Error accumulation using homography. (a) Rendered mosaic. (b) Projected key-frame borders. The last key-frame is the most deformed.

$$E(S) = E_C(S) + \lambda E_{Sm}(S), \tag{2}$$

where E_C is the correspondence energy function and E_{Sm} is the smoothness energy. The constant λ balances the compromise between precision and mesh smoothness. The registration is solved by finding the mesh S which minimizes $E(S)$. The correspondence energy is proportional to the projection error of warped features, while the smoothness energy measures the discontinuities on S; this energy is important to remove outlier feature matchings. However, the initial formulation described by Eq. (2) was designed for pairwise image registration only. Registration of sequences of images poses some additional problems. If only pairwise registration is used to align a sequence of images, over-deformation may occur due to error accumulation (**Fig. 3**).

To avoid error accumulation, a modified version of the previous energy function is presented. The new term, $E_{Ref}(S - S_{Ref})$ is named *reference mesh energy*. The mesh S_{Ref} represents a model of how the mesh S should look like without over-deformation. Alternatively, it is how the user of the mosaic system would expect the image (warped by S) to look like. The constant μ regulates the reference mesh energy weight. The new formulation is presented below:

$$E'(S) = E_C(S) + \lambda E_{Sm}(S) + \mu E_{Ref}(S - S_{Ref}). \tag{3}$$

The next sections present the energy functions in more detail.

3.3.3 Correspondence Energy

The correspondence energy $E_C(S)$ is a function of the projection error of the matched features. The matched feature set is represented by M. The matched feature pair $c \in M$ is composed of two features (c_0, c_1), where c_0 is a feature found in the target image and c_1 is a feature found in the image being warped. Let $w(c_1, S)$ be the warping function. The function υ is the same robust estimator used by Zhu et al. [8]. Its definition is given below:

$$\upsilon(\delta, \sigma) = \begin{cases} \frac{\|\delta\|^2}{\sigma^\gamma} & \text{if } \|\delta\| \leq \sigma \\ \sigma^{2-\gamma} & \text{otherwise} \end{cases} \tag{4}$$

$$E_C(S) = \sum_{c\in M} \upsilon(c_0 - w(c_1, S), \sigma) \tag{5}$$

The function υ has two parameters: the projection error δ and the radius of tolerance σ. The matches whose projection error is greater than the radius of tolerance are considered outliers and penalized. The radius of tolerance σ dictates which matched feature pairs will be considered outliers. The objective of σ is to remove outliers from the energy calculation.

3.3.4 Smoothness Energy

The correspondence energy, if used alone, can not handle a great number of outliers among the matched features. Also, since the non-rigid deformation model is high-dimensional, the optimization method may very easily get trapped in a local optimum. A smoothness constraint is added to the model in order to avoid these problems. The proposed method uses the same smoothness constraint found in Zhu et al. [8] and Pilet et al. [2]. It is explained as follows.

The smoothness energy E_{Sm} of the mesh S is the sum of the approximate second derivative of S. Let E be the set of all collinear control points in S that define two adjacent edges. The definition of E_{Sm} is given below:

$$\begin{aligned} E_{Sm}(S) &= \textstyle\sum_{i,j,k\in E}(-x_i + 2x_j - x_k)^2 + (-y_i + 2y_j - y_k)^2 \\ &= X^{\mathrm{T}}KX + Y^{\mathrm{T}}KY, \end{aligned} \tag{6}$$

where $K = K'^{\mathrm{T}}K'$, and K' is a matrix containing one row per triplet in E and one column per mesh vertex. The matrix K' gives the matrix form of the terms $(-x_i + 2x_j - x_k)$ and$(-y_i + 2y_j - y_k)$. The row corresponding to the triplet (i, j, k) has all of its values zero except in columns i, j, and k, that have values -1, 2, and -1, respectively.

3.3.5 Reference Mesh Energy

The registration using the energy function in Eq. (2) is only suited for pairwise registration, because alignment error may accumulate, as shown in Fig. 3. The role of the reference mesh energy is to alleviate this problem. This energy is proportional to the L_2 distance between the mesh S and the reference mesh S_{Ref}. The former is the registration solution and the latter is an approximation of how S should be if it has no over-deformation. The criteria selected to generate S_{Ref} was to make S_{Ref} look similar to the original captured image. S_{Ref} is defined as the similarity transformation (i.e., rotation, translation and scaling) that minimizes the correspondence energy. This mesh can be computed efficiently by reducing the projection error using the similarity transformations combined with RANSAC. The reference mesh energy is defined below:

$$E_{Ref}(S - S_{Ref}) = \|S - S_{Ref}\|^2. \tag{7}$$

During the optimization process, the reference mesh energy is stronger in the regions of the mesh S where there are no features. While the region with features is deformed to minimize the projection error, the region without features is deformed by similarity transformations. These local differences in the deformation are not possible for global registration models.

3.3.6 Optimization Routine

In this section we will demonstrate that, as pointed in Ref. [8], the projection error δ can be solved as a sparse linear system. Let c_0 and c_1 be matched feature coordinates belonging to a pair of images being registered. They are defined as $c_0 = (c_{0x}, c_{0y})$ and $c_1 = (c_{1x}, c_{1y})$. The feature c_1 belongs to the image being warped. Let N be the number of control points in the mesh. The feature c_1 lies inside the triangle defined by the control points $v_i, v_j, v_k \in S_0$, (calculated regarding the identity mesh). The indexes i, j, and k are in the range $[1, N]$. Also, let $t_{c_1} \in R^N$ be a vector representing the barycentric coordinates of the feature point c_1. The vector t_{c_1} has all its values 0, except in the coordinates i, j, and k, where the barycentric coordinates of c_1 in relation to v_i, v_j, and v_k are set, respectively. With these definitions, the projection error can be defined as:

$$\|\delta\|^2 = (c_{0x} - t_{c_1}^{\mathrm{T}}X)^2 + (c_{0y} - t_{c_1}^{\mathrm{T}}Y)^2, \tag{8}$$

where X and Y are the coordinates of the mesh control points. The Eq. (8) can be expanded as:

$$\begin{aligned} \|\delta\|^2 = {} & c_{0x}^2 + c_{0y}^2 - 2(c_{0x}t_{c_1}^{\mathrm{T}}X + c_{0y}t_{c_1}^{\mathrm{T}}Y) \\ & + X^{\mathrm{T}}t_{c_1}t_{c_1}^{\mathrm{T}}X + Y^{\mathrm{T}}t_{c_1}t_{c_1}^{\mathrm{T}}Y \end{aligned} \tag{9}$$

Using Eqs. (6), (7), and (9), the energy $E'(S)$ in Eq. (3) can be rewritten as:

$$\begin{aligned} & E'(S) = \\ & \frac{1}{\sigma^\gamma}\sum_{c\in M_{Inl}}\left(c_{0x}^2 + c_{0y}^2 - 2\begin{bmatrix} c_{0x}t_{c_1} \\ c_{0y}t_{c_1} \end{bmatrix}^{\mathrm{T}} S\right) + \\ & \frac{1}{\sigma^\gamma}\sum_{c\in M_{Inl}}\left(S^{\mathrm{T}}\begin{bmatrix} t_{c_1}t_{c_1}^{\mathrm{T}} & 0 \\ 0 & t_{c_1}t_{c_1}^{\mathrm{T}} \end{bmatrix} S\right) + \\ & |M_{Out}|\sigma^{2-\gamma} + \lambda(X^{\mathrm{T}}KX + Y^{\mathrm{T}}KY) + \mu\|S - S_{Ref}\|^2, \end{aligned} \tag{10}$$

where M_{Inl} is the set of inlier matches, M_{Out} is the set of outlier matches. The following definitions are done for simplification: $A = \frac{1}{\sigma^\gamma}\sum_{c\in M_{Inl}} t_{c_1}t_{c_1}^{\mathrm{T}}$, and $b = \begin{bmatrix} b_x \\ b_y \end{bmatrix} = \frac{1}{\sigma^\gamma}\sum_{c\in M_{Inl}}\begin{bmatrix} c_{0x}t_{c_1} \\ c_{0y}t_{c_1} \end{bmatrix}$. Computing the gradient of E' and setting it to zero, the mesh S can be found by solving a linear system:

$$S = \begin{bmatrix} \lambda K + A + \mu I & 0 \\ 0 & \lambda K + A + \mu I \end{bmatrix}^{-1}(b + \mu S_{Ref}). \tag{11}$$

The optimization routine is summarized in Algorithm 1.

The optimization is repeated varying the value of σ from σ_0 to $\varepsilon_{ProjErr}$. At each iteration, σ is multiplied by η, a real value in the range $(0, 1)$. At the beginning, σ is large, allowing many possible outliers to influence the result of the optimization process. However, since the module of the derivative of the E_C is small when σ is large, E_{Sm} and E_{Ref} have a larger weight and they initially guide the optimization. As the value of σ decreases, the weight of E_C increases, guiding the optimization to minimize the projection error of the remaining inliers. In this way, this registration method is robust to outliers. The process stops when σ is smaller than $\varepsilon_{ProjErr}$.

3.4 Triangle-wise Graph Cut

It is generally impossible to guarantee that the alignment of all regions of an image will be equally good. This is specially true in the particular case of non-rigid registration, where regions with few features are often misaligned. For this reason, the keyframes must be carefully stitched into the mosaic. The standard approaches for this problem consist of selecting which pixels of the new image will be used in the mosaic. This is generally done

```
input : $S_0, \lambda, \mu, \sigma_0, \eta, M$
output: S = { X, Y}

Pre-compute $\lambda K, \mu I$
Pre-compute $S_{Ref} = (X_{Ref}, Y_{Ref})$ using $M$
for $c = (c_0, c_1) \in M$ do
    Pre-compute the baricentric coordinates of $c_0$
end
$\sigma \leftarrow \sigma_0$
while $\sigma > \varepsilon_{ProjErr}$ do
    Compute $A, b$
    $X \leftarrow [\lambda K + A + \mu I]^{-1}(b_x + \mu X_{Ref})$
    $Y \leftarrow [\lambda K + A + \mu I]^{-1}(b_y + \mu Y_{Ref})$
    $\sigma \Leftarrow \eta\sigma$
end
```

Algorithm 1: Image registration algorithm. The parameter S_0 is the identity mesh, $\varepsilon_{ProjErr}$ is the error tolerance for inlier matchings, λ is the smoothness strength parameter, μ is the reference mesh similarity parameter, M is the set of matched features, σ_0 is the initial radius of tolerance, and η is the radius of tolerance decay rate. The parameter values are shown in Section 4.1.

by defining a stitching line. Pixels in one side of the stitching line are added to the mosaic, and the pixels in the other side are ignored. A common criteria for selecting the stitching line is to make it pass through pixels which are well aligned to the mosaic. By this, the pixels in both sides of the stitch (the pixels of the new registered key-frame and the pixels already in the mosaic) will be similar and the final composite mosaic will be seamless. The methods in Ref. [13], for example, use this approach. However, pixel selection is too slow for a real time method, specially dealing with high resolution images.

Our proposed solution to this problem is to select mesh triangles instead of pixels. Even in a mesh with a high degree of freedom, there are much fewer triangles than pixels in the key-frame. In our approach, we define a stitching line that passes through triangles that are well aligned. The number of inlier features (i.e., feature points correctly aligned to the mosaic) inside a triangle was used to evaluate its alignment. It was observed that triangles with inlier features inside are much better aligned than triangles without them. The stitching line is selected by solving a graph cut formulation, which is presented below.

Let $S_0, S_1, S_2, ..., S_{n-1}$ be the meshes already added to the mosaic. They are represented in gray in **Fig. 4** (a). Let $T_0, T_1, T_2, ..., T_{n-1}$ be the set of triangles that compose these meshes. Each one of these sets is defined as $T_i = \{\tau_i^0, \tau_i^1, ..., \tau_i^m\}$, where m is the number of triangles inside each mesh (m is constant). Each triangle is defined by three non-identical control points: $\tau_i^j = \{v_a, v_b, v_c\}; v_a, v_b, v_c \in S_i$. Now, let $T'_0, T'_1, T'_2, ..., T'_{n-1}$ be the set of triangles, for all previously added meshes, which were selected to be included into the mosaic. Therefore, $T'_i \subseteq T_i$.

Let S_n be the next mesh to be inserted into the mosaic (represented in red in Fig. 4 (a), and C_n the matched features of the last key-frame F_n. Let $\omega : T_n \rightarrow \mathbb{N}$ be a function that receives as parameter a triangle τ_n^j from the mesh S_n and returns the number of aligned features which lie inside τ_n^j.

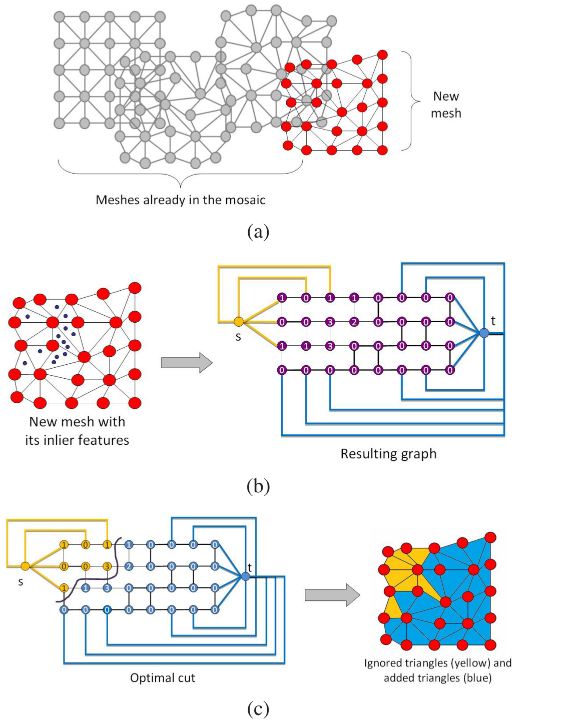

New mesh
Meshes already in the mosaic
(a)
New mesh with its inlier features
s
t
Resulting graph
(b)
s
t
Optimal cut
Ignored triangles (yellow) and added triangles (blue)
(c)

Fig. 4 Proposed triangle-wise graph cut algorithm. (a) New mesh being added into the mosaic. (b) A graph is created to represent the new mesh; the vertices representing the border triangles which overlap with the mosaic receive a $\mathfrak{s}$ edge and the other vertices representing border triangles receive a $\mathfrak{t}$ edge; the weight of each vertex is inversely proportional to the number of aligned features inside its corresponding triangle. (c) The minimum cut is computed; the triangles whose vertices are in the side of s will not be added to the mosaic, while the other triangles will.

To use graph cut, a graph and an edge-weight function must be defined. The graph $\{U, E\}$ is constructed based on S_n. The set of vertices U has one vertex u_j for each triangle τ_n^j in S_n. The set of edges E has one edge for each pair of adjacent triangles in S_n (two triangles are adjacent if they share an edge). To follow the graph cut formulation, two special vertices, $\mathfrak{s}$ and $\mathfrak{t}$ (source and sink) are added. Let Tb_n be the set of triangles that are on the borders of S_n. For each one of the triangles in Tb_n one extra edge is added to E: if a triangle $\tau_b \in Tb_n$, represented in U by the vertex u_j, has at least one intersection with one of the triangles already in the mosaic, the edge $(\mathfrak{s}, u_j)$ is added. These triangles will definitely not be added to the mosaic. Otherwise, the edge $(u_j, \mathfrak{t})$ is added to E. It means that the border triangles that do not intersect the mosaic will definitely be added to the mosaic. Now let $w : E \rightarrow \mathbb{R}$ be an edge-weight function. This function is defined as follows:

$$w(u_i, u_j) = \begin{cases} (\omega(\tau_n^i) + \omega(\tau_n^j) + \varepsilon)^{-1} & u_i \neq \mathfrak{s}, u_j \neq \mathfrak{t} \\ \infty & \text{otherwise} \end{cases} \qquad (12)$$

where ε is a small value used to avoid divisions by zero. This function gives weights that are inversely proportional to the number of inlier features inside the adjacent triangles, enforcing that the minimum cut must pass through the triangles with the most inlier features. The edges from the sink and source vertices are given infinite weight.

The graph generated by this process is illustrated in Fig. 4 (b). Using these definitions, the optimal stitching can be computed using the max-flow min-cut algorithm as defined in Ref. [16], for example. The triangles τ_n^j whose vertices u_j end up in the side of $\mathfrak{s}$ (regarding the optimal cut) are not added to the mosaic. The other triangles are selected and added to T'_n. Figure 4 (c) shows the minimum cut obtained by the graph-cut algorithm.

4. Experiments

The objective of the experiments is to demonstrate four points: the proposed method has a smaller projection error comparing to the classical approaches, the mosaics created by the proposed method have less over-deformation, the proposed method can run in real-time, and that the results obtained by the proposed method are more robust than the results obtained by classical approaches in the kind of video considered.

4.1 Experimental Setup

The project was run in a computer with Intel(R) Core(TM) i7 CPU (2.93 GHz) and 4 GB of RAM. The proposed method was implemented using the OpenCV library. The parameter setting is presented in **Table 1**.

For the reference mesh computation, the precision of *RANSAC* is set to 99% in the presence of 70% of outliers. The size of the mesh was 19 × 28 control points. The videos used on the experiments had a resolution of 720 × 480.

4.2 Registration Precision

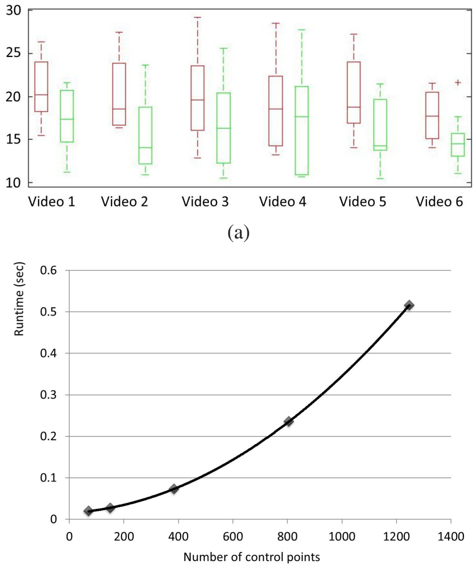
This experiment presents the comparison between homography and non-rigid transformations concerning precision by means of mean appearance error, defined as the mean absolute difference between all aligned pixels. The experiments were conducted by registering pairs of images. **Figure 5** (a) shows the results of the average error of pair-wise registration over different video sequences. **Figure 6** shows a detail of a pair of registered key-frames (the averaged image). As can be seen, the results achieved by the registration procedure used by the proposed method are always more precise than the results using homography. This happens because the deformation field between the pairs of images can not be precisely described by a global transformation like projection, since the displacement field depends on the geometry of the scene.

4.3 Over-deformation Avoidance

This set of experiments compares mosaics done by the proposed method and non-rigid registration as described by Ref. [8]. The comparisons are done regarding over-deformation. **Figure 7** shows the results. Both methods use the same set of key-frames. As previously shown in Fig. 3, using homography, the registration error tends to build up and cause the key-frames to over-deform. When using only non-rigid registration, without the reference mesh energy, error accumulation also happens, even though the alignment error is smaller compared to homography. The proposed method, using the reference mesh energy, minimizes the amount of over-deformation. This result may be achieved by related methods using bundle adjustment, but the proposed method achieves the same by only doing pair-wise registration.

Table 1 Parameter settings for the proposed method.

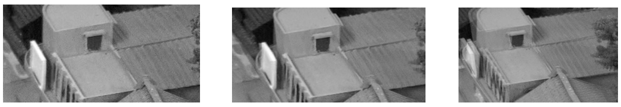
Description	Param.	Value
Key-frame selection threshold.	ϑ	0.4
Key-frame selection weight function std. deviation.	ς	1.0
Feature matching epipolar constraint threshold	ε_{Match}	10^{-2}
Smoothness energy parameter.	λ	10^{-6}
Reference mesh energy parameter.	μ	10^{-4}
Correspondence energy parameter.	γ	4
Registration parameter; initial radius of tolerance.	σ_0	32
Minimum radius of tolerance; i.e., projection error.	$\sigma_{\min}$	3
Radius of tolerance decay rate.	η	0.5



(a)

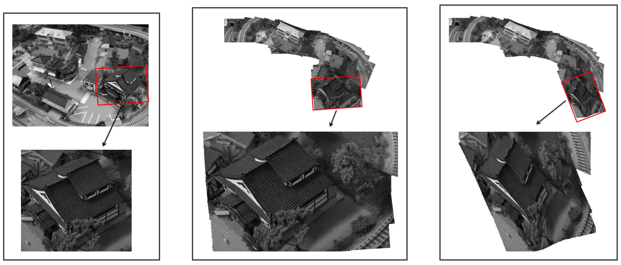
(b)

Fig. 5 (a): Appearance error with homography and non-rigid transformations. The error is measured as the mean absolute difference between pixel gray-scale values of aligned pixels, in a set of videos. The red boxes show the results obtained by homography, and the green boxes represent the results of the proposed method; (b): Execution time (seconds) in relation to number of control points.



(a) (b) (c)

Fig. 6 Detail of a pair of registered key-frames, showing the average of the superposition of the key-frames. (a) Original video key-frame. (b) Key-frame aligned by the proposed method. (c) Key-frame aligned by homography.



(a) (b) (c)

Fig. 7 Mosaicing results, regarding over-deformation. (a) City model used in the experiments, showing the expected undeformed key-frame. (b) Results obtained by the proposed method. (c) Results obtained using only non-rigid registration without the reference mesh energy. The mosaic generated by the proposed method presents less over-deformation.

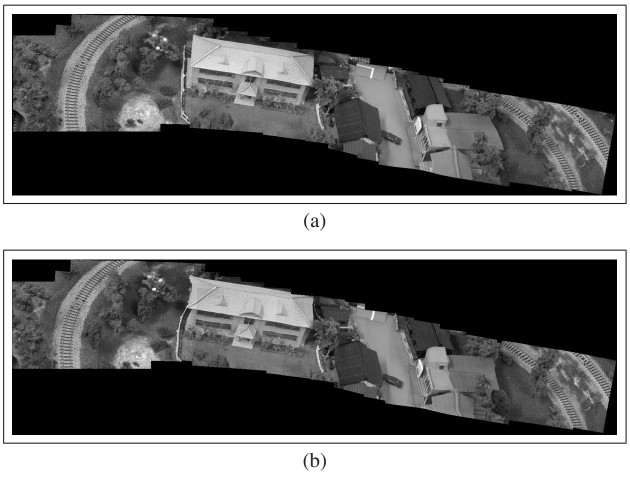


Fig. 8 Mosaic stitching results. (a) Results of the proposed image stitching method. (b) Results obtained by overlapping the selected key-frames. The mosaic generated by the proposed method presents much less seam marks.

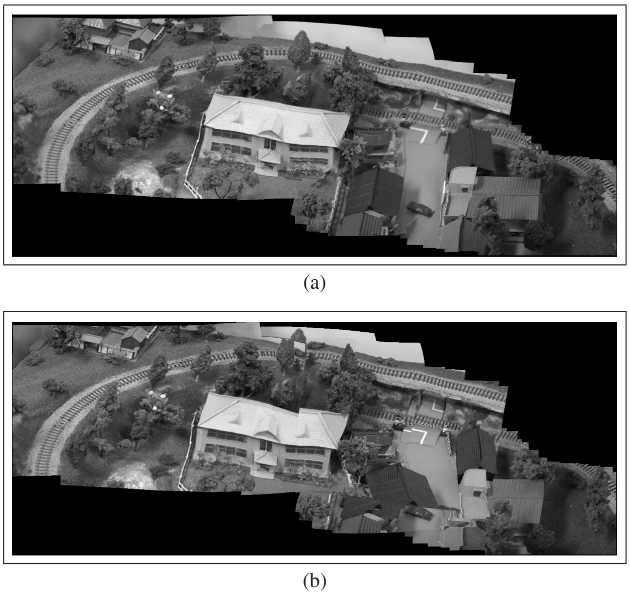


Fig. 9 Mosaic stitching results. (a) Results of the proposed image stitching method. (b) Results of just overlapping the selected key-frames. The mosaic generated by the proposed method presents much less seam marks.

4.4 Mosaic Stitching

This set of experiments is done to evaluate the improvements in the final mosaic when applying the triangle-wise graph cut algorithm presented in Section 3.4. The comparisons were done using two input videos. The proposed stitching method was compared to the mosaics created overlapping consecutively selected key-frames. Both results use the same set of input images and the same registered feature points. **Figure 8** shows the results from the first video, and **Fig. 9** shows the results for the second video. Figure 8 (a) and Fig. 9 (a) show the results of the proposed method, while Fig. 8 (b) and Fig. 9 (b) show the results of mosaic created by overlapping the key-frames. **Figure 10** and **11** show details of these mosaics. **Figure 12** shows the triangles selected by the proposed method to be included into the final mosaics. As can be seen in these results, the proposed stitching scheme can ignore most registration errors which occur in regions without inlier matched features.

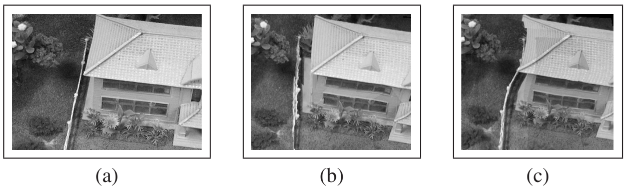


Fig. 10 Details of the mosaic in Fig. 8. (a) Video key-frame. (b) Result of the proposed stitching method. (c) Result of overlapping the key-frames.

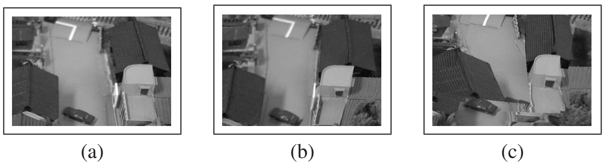


Fig. 11 Details of the mosaic in Fig. 9. (a) Video key-frame. (b) Result of the proposed stitching method. (c) Result of overlapping the key-frames.

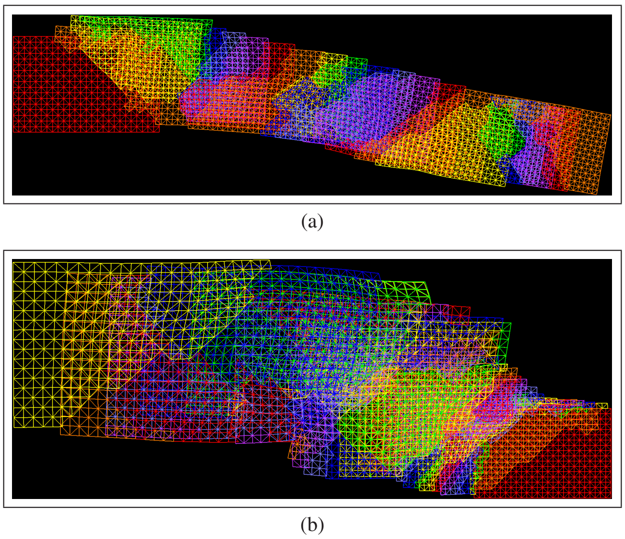


Fig. 12 Triangles selected by the proposed stitching method in (a) Fig. 8 and (b) Fig. 9.

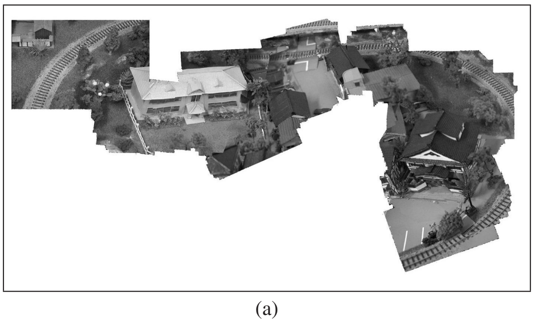
4.5 Comparison with a Standard Method

In this set of experiments, the proposed method was compared to a standard method, implemented by Microsoft Image Composite Editor (ICE), version 1.3.5. Using ICE, the user can choose different camera movements. The one which yielded the best result was selected. The proposed method used the parameters described in Section 4.1. ICE and the proposed method used the same set of key-frames. **Figure 13** shows the mosaic created from a video taken by a camera moving over a city model.

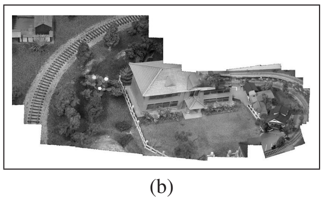
Figure 13 (a) shows the results obtained by the proposed method. Figure 13 (b) shows the results obtained by ICE. As can be seen, the results obtained by the proposed method are more complete than the results given by ICE. This happens because of the complex camera movement and the non-planar surface, which violate the homography constraints used by ICE.

4.6 Computational Complexity

The current implementation of the proposed method runs in about 32 frames per second with a tax of 2 key-frames selected per second, what is reasonable for videos where the camera movement is not excessively fast.



(a)



(b)

Fig. 13 Comparison between the proposed method and a standard method. (a) Results of the proposed method. (b) Results of a standard method (Microsoft ICE). The proposed method created a more complete mosaic since it can handle more complex camera movements.

Each iteration of key-frame selection takes approximately 0.031 seconds, about 32 frames per second, enough for most videos. Figure 5 (b) shows runtime regarding only the registration procedure. It was executed 10 times for each quantity of control points (the computation of the reference mesh is included). As can be seen from the experiments, registration runtime grows slowly. This happens because of the implementation that uses sparse matrices to represent the registration model. The runtime of the key-frame selection and mosaic creation procedure were also computed. Using approximately 1,000 triangles, the registration can be done in about 3 frames per second. Regarding mosaic creation, each key-frame takes on average 0.28 seconds to be stitched by graph cut and rendered, a tax of nearly 3 frames per second.

The conclusion is that the proposed method can run in real time, given that the frame selection rate is about 3 frames per second, reasonable when camera movement is not too fast. Further optimization on the method may be performed in the future.

5. Conclusions

This paper presented a new mosaicing technique based on feature based non-rigid registration. The proposed method can be used to create mosaics of non-planar surfaces in real-time. This model deals with the problem of over-deformation using only pairwise registration, and creates mosaics with smaller alignment error compared to standard approaches. For this purpose, the reference mesh energy was presented. An efficient method of key-frame selection designed to achieve real-time performance was proposed. Also, a triangle-wise graph cut algorithm capable of reducing the error in the final mosaic was presented.

The proposed method has some restrictions. First, since there is no bundle adjustment, the generated mosaic is prone to error if a region of the scene is recorded twice (loop). This will require an efficient global registration method able to run in real-time. The proposed method also fails when sharp discontinuities in the optical flow are present, due to the mesh smoothness constraint. This limitation could be solved by detecting these discontinuities and segmenting the image into patches which can be aligned by a smooth warping function. This will be the target of our future research.

References

[1] Hsu, S., Sawhney, H.S. and Kumar, R.: Automated mosaics via topology inference, *IEEE Computer Graphics and Applications*, Vol.22, No.2, pp.44–54 (2002).

[2] Pilet, J., Lepetit, V. and Fua, P.: Real-time Non-rigid Surface Detection, *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pp.822–828 (2005).

[3] Sawhney, H.S., Hsu, S. and Kumar, R.: Robust video mosaicing through topology inference and local to global alignment, *Computer Vision ECCV 98, Lecture Notes in Computer Science*, Vol.1407, pp.103–119 (1998).

[4] Chaiyasarn, K., Kim, T.-K., Viola, F., Cipolla, R. and Soga, K.: Image mosaicing via quadric surface estimation with priors for tunnel inspection, *16th IEEE International Conference on Image Processing* (*ICIP*), *2009*, pp.537–540 (2009).

[5] Brown, M. and Lowe, D.: Automatic Panoramic Image Stitching using Invariant Features, *International Journal of Computer Vision*, Vol.74, pp.59–73 (2007).

[6] Can, A., Stewart, C.V., Roysam, B. and Tanenbaum, H.L.: A feature-based technique for joint, linear estimation of high-order image-to-mosaic transformations: Application to mosaicing the curved human retina, *Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2000*, Vol.2, pp.585–591 (2000).

[7] Crispell, D., Mundy, J. and Taubin, G.: Parallax-Free Registration of Aerial Video, *Proc. British Machine Vision Conf.* (2008).

[8] Zhu, J., Lyu, M.R. and Huang, T.S.: A Fast 2D Shape Recovery Approach by Fusing Features and Appearance, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.31, pp.1210–1224 (2009).

[9] Deng, Y. and Zhang, T.: Generating Panorama Photos, *Proc. SPIE Internet Multimedia Management Systems IV* (2003).

[10] Chui, H. and Rangarajan, A.: A new point matching algorithm for non-rigid registration, *Computer Vision and Image Understanding*, Vol.89, No.2-3, pp.114–141 (2003).

[11] Peleg, S., Rousso, B., Rav-Acha, A. and Zomet, A.: Mosaicing on adaptive manifolds, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.22, No.10, pp.1144–1154 (2000).

[12] Szeliski, R.: Image alignment and stitching: A tutorial, *Found. Trends. Comput. Graph. Vis.*, Vol.2, No.1, pp.1–104 (2006).

[13] Kwatra, V., Schodl, A., Essa, I.A., Turk, G. and Bobick, A.F.: Graphcut textures: Image and video synthesis using graph cuts, *ACM Transactions on Graphics*, Vol.22, No.3, pp.277–286 (2003).

[14] Bay, H., Tuytelaars, T. and Gool, L.V.: SURF: Speeded Up Robust Features, *Lecture Notes in Computer Science, Computer Vision ECCV 2006*, Vol.3951, pp.404–417 (2006).

[15] Hartley, R.I. and Zisserman, A.: Multiple View Geometry in Computer Vision, *International Journal of Computer Vision, Second Edition* (2004).

[16] Boykov, Y.Y. and Jolly, M.-P.: Interactive graph cuts for optimal boundary amp; region segmentation of objects in N-D images, *Proc. Eighth IEEE International Conference on Computer Vision, 2001, ICCV 2001*, Vol.1, pp.105–112 (2001).

[17] Lowe, D.G.: Distinctive image features from scale-invariant keypoints, *International Journal of Computer Vision*, Vol.60, No.2, pp.91–110 (2004).

Rafael Henrique Castanheira de Souza received, in 2006 and 2008, his B.S. and M.S. in Computer Science from the State University of Campinas, Campinas, Brazil. In 2008 he joined as research student the Graduate School of Science and Engineering at TITech. From 2009, he started his Ph.D. in the same institution.

Masatoshi Okutomi received a B.Eng. degree in Mathematical Engineering and Information Physics from the University of Tokyo, Tokyo, Japan, in 1981, and a M.Eng. degree in Control Engineering from Tokyo Institute of Technology, Tokyo, Japan, in 1983. He joined Canon Research Center, Canon Inc., Tokyo, Japan, in 1983. From 1987 to 1990, he was a Visiting Research Scientist in the School of Computer Science at Carnegie Mellon University, Pittsburgh, PA. In 1993, he received a Ph.D. degree for his research on stereo vision from Tokyo Institute of Technology. Since 1994, he has been with Tokyo Institute of Technology, where he is currently a Professor of the Department of Mechanical and Control Engineering, the Graduate School of Science and Engineering.

Akihiko Torii received his B.Eng., M.Eng., and Ph.D. in Information and Computer Sciences from Chiba University, Japan, in 2001, 2003, and 2006, respectively. He joined Czech Technical University in Prague, Czech Republic, as a Research Fellow from 2006 to 2010. Since 2010, he has been an Assistant Professor of the Department of Mechanical and Control Engineering, the Graduate School of Science and Engineering, Tokyo Institute of Technology, Japan.

(Communicated by *Shin'ichi Satoh*)