Applications of 2D Image Registration

F. Odone and A. Fusiello

Dept. of Computing & Electrical Engineering
Heriot-Watt University
Email: {fusiello,franci}@cee.hw.ac.uk

Abstract

This report outlines the work carried out by the authors on 2D image registration. Starting from the background theory, we will show several application ranging from mosaics to content-based video manipulation. In particular we will focus on underwater video mosaicing. No assumptions are made about the video source, as our algorithm can cope with data coming from disparate devices, such underwater cameras and commercial hand-held cameras. A wide set of results is shown.

1 Introduction

Image registration[Bro92, GM99] is the process of determining correspondence between all points into images of the same scene. By registering two images, information from different sources can be combined, the geometry of the scene can be recovered, and changes occurred in the scene between the times the images there obtained can be determined. Image registration is often needed in medical image analysis, processing of remotely sensed data, robot vision, automated monitoring, and industrial inspection.
Direct minimization of pixel intensity differences has been widely used to register images [Sze96, IAB$^+$96, SA96]. This is closely related to the problem of approximating the *2D motion field* [BFB94, CV92], that is, the vector field that describes the relative motion between the viewing camera and the observed scene. An approximation of 2D motion field is the optical flow. Flow-based methods, though dense and accurate, are computationally expensive, and are very sensitive to local minima. An alternative approach consists of selecting a number of features from the images, establishing correspondences between them, and determining a transformation function that maps points in one image to points in the other image.
We hold that, although less common, feature based registration [ZFD97] is to be preferred. Besides, being less complex, methods based on the tracking of two dimensional features (such as corners) use motion information only where it is most reliable, because feature points do not suffer from the *aperture effect*[TV98], typical of an optical flow approach. Digital video sequences have a very high frame rate (usually 25 frames per second), which strongly points toward a feature tracking based registration, because on one hand it requires a fast registration and on the other it makes tracking feasible, being dense.
Among the various applications of image registration, we will focus mainly on *Mosaicing*, that is the automatic alignment of multiple images into larger aggregates [Sze96]. In many real scenarios, it is adequate to describe a static scene, the image motion of which is only due to camera motion, using a mosaic.

A mosaic description suffices when camera center does not move appreciably or when image motion can be well approximated by that of a single plane. In both cases, a two-dimensional (2D) motion model can be adopted.
Other works in mosaicing includes [CZ98, RPFRA98, Dav98, GSV98, MC97, SK97].

The structure of this report is the following: section 2 reviews the background notions needed, in section 3 the feature tracker used to extract and follow the interesting points is described, section 4 reports how the transformation between two images is computed. Section 5 deals with the motion estimation problem. Section 6 gives a brief introduction to mosaicing techniques. Section 7 specializes to the stabilization problem, while section 8 reports a mosaic-based approach to motion segmentation. In section 9 an interesting application of mosaic-based segmentation can be found. Section 10 contains a set of results obtained with our experiments in mosaics construction, stabilization of sequences and video coding. The report also contains three appendices that give more details about the two view geometry, the metric rectification and the information that can be extracted from homographies.

2 Background

A non-singular linear transformation of the projective plane [SK52] into itself is called *homography* (or *collineation*). The most general homography is represented by a non-singular 3×3 matrix $\mathbf{H}$:

$$\lambda \begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = \begin{bmatrix} H_{1,1} & H_{1,2} & H_{1,3} \\ H_{2,1} & H_{2,2} & H_{2,3} \\ H_{3,1} & H_{3,2} & H_{3,3} \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}. \tag{1}$$

Points are expressed in homogeneous coordinates, that is, we denote 2-D points in the image plane as $\tilde{\mathbf{m}} = (x_1, x_2, x_3)$ with $\mathbf{m} = (u, v) = (x_1/x_3, x_2/x_3)$ being the corresponding Cartesian coordinates[1]. The matrix $\mathbf{H}$ has 8 degrees of freedom, being defined up to a scale factor. The transformation is linear in projective (or homogeneous) coordinates, but it is *non linear* in Cartesian coordinates:

$$\begin{cases} u' = \dfrac{H_{1,1}u + H_{1,2}v + H_{1,3}}{H_{3,1}u + H_{3,2}v + H_{3,3}} \\ v' = \dfrac{H_{2,1}u + H_{2,2}v + H_{2,3}}{H_{3,1}u + H_{3,2}v + H_{3,3}} \end{cases}. \tag{2}$$

Two images taken by a moving camera are related by a projective plane transformation in two cases: a planar scene imaged from different points of view or a 3D scene viewed from the same point of view (the camera is rotating around its optical centre).
In general, it can be seen that two points $\mathbf{m}$ and $\mathbf{m}'$, projection of the 3D point $\mathbf{w}$ onto the first and the second view respectively, are related by[2]

$$\kappa' \tilde{\mathbf{m}}' = \kappa \mathbf{A}' \mathbf{R} \mathbf{A}^{-1} \tilde{\mathbf{m}} + \mathbf{A}' \mathbf{t}. \tag{3}$$

where $\mathbf{A}$ is a 3×3 matrix encoding the *intrinsic* parameters of the camera (focal length, aspect ratio, image centre), $\mathbf{R}$ is a 3×3 rotation matrix which gives the camera rotation between the two views, and $\mathbf{t}$ is a 3×1 vector representing the translation of the optical centre between the two views. κ and κ' are the distances of the 3D point from the first and second camera focal planes.

[1]We shall henceforth use the symbol ˜to indicate homogeneous coordinates.
[2]See Appendix A for details.

If camera is rotating, then $\mathbf{t} = 0$ and we get:

$$\frac{\kappa'}{\kappa}\tilde{\mathbf{m}}' = \mathbf{A}'\mathbf{R}\mathbf{A}^{-1}\tilde{\mathbf{m}}. \tag{4}$$

The 3×3 matrix $\mathbf{H}_\infty = \mathbf{A}'\mathbf{R}\mathbf{A}^{-1}$ represents an homography, and does not depend on the 3D structure. In the other case, if the camera undergoes a general rigid motion, but 3D points lie on a plane Π with Cartesian equation $\mathbf{n}^\top \mathbf{w} = d$, Eq. (3) can be specialized, obtaining:

$$\frac{\kappa'}{\kappa}\tilde{\mathbf{m}}' = \left(\mathbf{H}_\infty + \frac{\mathbf{A}'\mathbf{t}\ \mathbf{n}^\top \mathbf{A}^{-1}}{d}\right)\tilde{\mathbf{m}}. \tag{5}$$

Therefore, there is a projective plane transformation between the two views induced by the plane Π, given by $\mathbf{H}_\Pi = \mathbf{H}_\infty + \mathbf{A}'\mathbf{t}\frac{\mathbf{n}^\top}{d}\mathbf{A}^{-1}$. The $\mathbf{H}_\infty$ homography, obtained in the previous case, can be interpreted as the homography induced by a very special plane, *the infinity plane*, as can be seen by letting $d \to \infty$ in (5).

It might be worth showing how two views are related in the general case of full 3D scene and arbitrary camera motion. Starting again from Eq.(3)

$$\kappa'\tilde{\mathbf{m}}' = \kappa\mathbf{H}_\infty\tilde{\mathbf{m}} + \mathbf{A}'\mathbf{t}, \tag{6}$$

and substituting $\mathbf{H}_\infty = \mathbf{H}_\Pi - \mathbf{A}'\mathbf{t}\frac{\mathbf{n}^\top}{d}\mathbf{A}^{-1}$, we obtain (from Eq. 5):

$$\frac{\kappa'}{\kappa}\tilde{\mathbf{m}}' = \mathbf{H}_\Pi\tilde{\mathbf{m}} + \mathbf{A}'\mathbf{t}\left(\frac{a}{d\ \kappa}\right). \tag{7}$$

where $a = d - \mathbf{n}^\top\kappa\mathbf{A}^{-1}\tilde{\mathbf{m}}$ is the orthogonal distance of the 3D point $\mathbf{w}$ (of which $\mathbf{m}$ and $\mathbf{m}'$ are projections) to the plane Π. If $\mathbf{w}$ is on the 3D plane Π, then $\tilde{\mathbf{m}}' \simeq \mathbf{H}_\Pi\tilde{\mathbf{m}}$. Otherwise, the remaining displacement, called *parallax*, is proportional to the *relative affine structure* $\gamma = a/(d\ \kappa)$ of $\mathbf{w}$ (wrt the plane Π) [SN96]. The relative affine structure of a point depends on its depth, on the choice of the first view and on the reference plane. When the reference plane is the plane at infinity, the relative affine structure reduces to $\gamma = 1/\kappa$, as can easily be seen from Eq.(3).

This is not all the story. The homography matrix $\mathbf{H}_\Pi$ that one can measure from image correspondences is defined only up to a scale factor. Since (in Eq. 7) it occurs in a sum, we need to normalize it.

Let consider a point $\mathbf{w}_0 \notin \Pi$ and scale $\mathbf{H}_\Pi$ to satisfy the equation

$$\tilde{\mathbf{m}}_0' \simeq \mathbf{H}_\Pi\tilde{\mathbf{m}}_0 + \mathbf{e}'. \tag{8}$$

This implies that $\frac{a_0}{d\ \kappa_0} = 1$, hence $d = \frac{a_0}{\kappa_0}$. Therefore, Eq. 7 rewrites:

$$\tilde{\mathbf{m}}' \simeq \mathbf{H}_\Pi\tilde{\mathbf{m}} + \mathbf{e}'\left(\frac{a}{\kappa}\frac{\kappa_0}{a_0}\right). \tag{9}$$

where $\mathbf{H}_\Pi$ is scaled to satisfy $\tilde{\mathbf{m}}_0' \simeq \mathbf{H}_\Pi\tilde{\mathbf{m}}_0 + \mathbf{e}'$. The relative affine structure is the product of two ratios, the first being the ratio of the perpendicular distance a of a point $\mathbf{w}$ to the plane Π and the depth κ to the reference camera, and the second ratio is of the same form but applied to a fixed point $\mathbf{w}_0$ which is used to set a uniform scale.

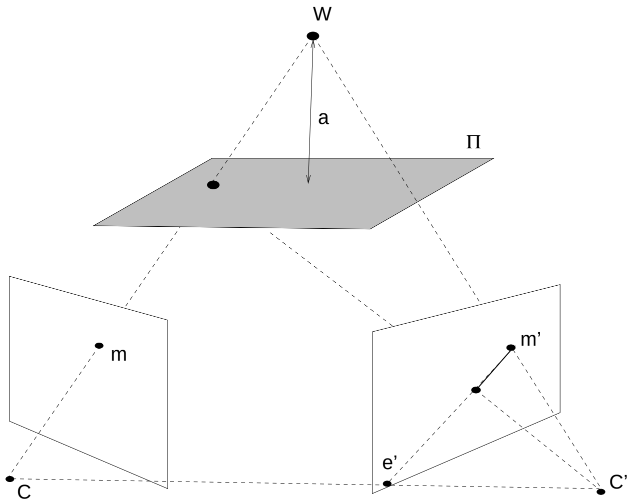

Figure 1: Geometric interpretation of the homography+parallax decomposition of Equation (7).

3 Homography computation

Let us suppose that we are given an image sequence in which there is a negligible parallax (i.e., subsequent frames are approximately related by a homography, as discussed in Sec. 2) and that point correspondences through the image sequence have been obtained by features tracking (Sec. 4). In this section we shall see how homographies are computed, and how to cope with moving objects. Four points (provided that no three of them are collinear) determine a unique homography. Indeed, eight independent parameters are required to define the homography. Each point correspondence in the plane provides two equations:

$$\begin{cases} u'(H_{3,1}u+H_{3,2}v+H_{3,3}) = H_{1,1}u+H_{1,2}v+H_{1,3} \\ v'(H_{3,1}u+H_{3,2}v+H_{3,3}) = H_{2,1}u+H_{2,2}v+H_{2,3} \end{cases} . \tag{10}$$

It is then necessary to find (at least) four point correspondences to define the transformation matrix uniquely (up to a scale factor). There are two methods of dealing with the unknown scale factor in a homogeneous matrix: choose one of the matrix elements to have a certain value, usually $H_{3,3} = 1$, or solve for the matrix up to a scale. We used the latter, which is more general. Equation (10) can be rearranged as:

$$\begin{bmatrix} u & v & 1 & 0 & 0 & 0 & -uu' & -vu' & -u \\ 0 & 0 & 0 & u & v & 1 & -uv' & -vv' & -v' \end{bmatrix} \begin{bmatrix} H_{1,1} \\ H_{1,2} \\ H_{1,3} \\ H_{2,1} \\ H_{2,2} \\ H_{2,3} \\ H_{3,1} \\ H_{3,2} \\ H_{3,3} \end{bmatrix} = \mathbf{0}. \tag{11}$$

For $n \geq 4$ points, we obtain a rank deficient system of homogeneous linear equations, which has the form $\mathbf{Lh} = \mathbf{0}$. If $n > 4$ there are more equations than unknown, and, in general, only a Least

Squares solution can be found. Let $\mathbf{L} = \mathbf{U}\mathbf{D}\mathbf{V}^\top$ be the Singular Value Decomposition (SVD) [GL96] of $\mathbf{L}$. One Least Squares solution is the column of $\mathbf{V}$ corresponding to the least singular value of $\mathbf{L}$. The computational cost of SVD is $O(n^3)$.
As pointed out by Hartley in the case of the Fundamental matrix estimation, a better conditioned problem is obtained by data *standardization*[Har95]. The points are translated so that their centroid is at the origin and are then scaled so that the average distance from the origin is equal to $\sqrt{2}$. Let $\mathbf{T}$ and $\mathbf{T}'$ the resulting transformation in the two images and $\tilde{\mathbf{m}}^* = \mathbf{T}\tilde{\mathbf{m}}$, $\tilde{\mathbf{m}}'^* = \mathbf{T}'\tilde{\mathbf{m}}'$ the transformed points. Using $\tilde{\mathbf{m}}^*$ and $\tilde{\mathbf{m}}'^*$ in the homography estimation algorithm, we obtain a matrix $\mathbf{H}^*$ that is related to the original one by $\mathbf{H}^* = \mathbf{T}'\mathbf{H}\mathbf{T}^{-1}$, as it can be easily seen.

3.1 Dominant motion estimation

In the case of a static scene with a moving camera, the Least Squares estimate are accurate enough. In presence moving objects the number outliers in the regression problem increases, since each pixel of a moving object is an outlier. Therefore, in this case, a robust method must be employed in order to estimate the homography that explains the motion of the *majority* of the features, that is the *dominant motion*. Unless the scene is cluttered with many moving objects, this is usually the motion of the camera with respect to the static background.
Least Median of Squares[RL87] is a robust regression technique which has been used in many Computer Vision applications[MMKR91, Zha97]. The principle behind LMedS is the following: given a regression problem, where d is the minimum number of points which determine a solution (four, in our case), compute a candidate model based on a randomly chosen d-tuple from the data; estimate the fit of this model to *all* the data, defined as the median of the squared residuals, and repeat optimizing the fit. The residuals are defined, in our case, for each point correspondence, as the distances between the warped and the actual point in the second image. In formulae, let $\hat{\mathbf{H}}$ be an approximate solution of (11), then the residuals are

$$s_j = ||\mathbf{m}'_j - \hat{\mathbf{H}}\mathbf{m}_j|| \quad j = 1 \dots n \tag{12}$$

where n is the number of point correspondences.
The data points that do not belong to the optimal model, which represent the majority of the data, are *outliers*. The *breakdown point*, i.e., the smallest fraction of outliers that can yield arbitrary estimate values, is 50%. In principle all the d-tuples should be evaluated; in practice a Monte Carlo technique is applied, in which only a random sample of size m is considered. Assuming that the whole set of points may contain up to a fraction ϵ of outliers, the probability that at least one of the m d-tuple consist of d inliers is given by

$$P = 1 - (1 - (1 - \epsilon)^d)^m. \tag{13}$$

Hence, given d, ϵ, and the required P (close to 1), one can determine m:

$$m = \frac{\log(1 - P)}{\log(1 - (1 - \epsilon)^d)}. \tag{14}$$

In our implementation we assume $\epsilon = 0.5$, and require $P = 0.99$, thus $m = 72$.
When Gaussian noise is present in addition to outliers, the relative statistical efficiency (i.e., the ratio between the lowest achievable variance for the estimated parameters and the actual variance) of the LMedS is low; to increase the efficiency, it is advisable to run a weighted LS fit after LMedS, with weights depending on the residual of the LMedS procedure [RL87].

The residuals s_j, $j = 1, \ldots, n$ are used to generate the weights for the final, weighted LS regression as follows. First, a robust standard deviation estimate [RL87] is computed as

$$\hat{\sigma} = 1.4826 \left(1 + \frac{5}{n-d}\right) \sqrt{\operatorname*{med}_j s_j^2}, \tag{15}$$

where d is the number of parameters (4 in our case). Second, a weight is assigned to each point correspondence, such that

$$w_j = \begin{cases} 1 & \text{if } |s_j|/\hat{\sigma} \leq 2.5, \\ 0 & \text{otherwise.} \end{cases} \tag{16}$$

The computational cost of LMedS with Monte Carlo speed up $O(mn \log n)$.

4 Feature tracking

In this section the Shi-Tomasi-Kanade tracker [ST94, TK91] will be briefly described. Consider an image sequence $I(\mathbf{m}, t)$, with $\mathbf{m} = [u, v]^\top$, the coordinates of an image point. If the time sampling frequency is sufficiently high, we can assume that small image regions are displaced but their intensities remain unchanged:

$$I(\mathbf{x}, t) = I(\delta(\mathbf{m}), t + \tau), \tag{17}$$

where $\delta(\cdot)$ is the *motion field*, specifying the *warping* that is applied to image points. The fast-sampling hypothesis allows us to approximate the motion with a translation, that is, $\delta(\mathbf{m}) = \mathbf{m} + \mathbf{d}$, where $\mathbf{d}$ is a displacement vector. The tracker's task is to compute $\mathbf{d}$ for a number of selected points for each pair of successive frames in the sequence.
As the image motion model is not perfect, and because of image noise, Eq. (17) is not satisfied exactly. The problem is then finding the displacement $\hat{\mathbf{d}}$ which minimizes the SSD residual:

$$\epsilon = \sum_{\mathcal{W}} \left[I(\mathbf{m} + \mathbf{d}, t + \tau) - I(\mathbf{m}, t) \right]^2 \tag{18}$$

where $\mathcal{W}$ is a small image window centered on the point for which $\mathbf{d}$ is computed. By plugging the first-order Taylor expansion of $I(\mathbf{m} + \mathbf{d}, t + \tau)$ into (18), and imposing that the derivatives with respect to $\mathbf{d}$ are zero, we obtain the linear system $\mathbf{G}\mathbf{d} = \mathbf{e}$, where

$$\mathbf{G} = \sum_{\mathcal{W}} \nabla I \, \nabla I^\top, \quad \mathbf{e} = -\tau \sum_{\mathcal{W}} I_t \nabla I, \tag{19}$$

with $= \nabla I = [\partial I / \partial u \; \partial I / \partial v]^\top$ and $I_t = \partial I / \partial t$. Using this linear approximation of the solution, the Newton-Raphson iterative algorithm for minimizing (18) writes:

$$\begin{cases} \mathbf{d}_0 = \mathbf{0} \\ \mathbf{d}_{k+1} = \mathbf{d}_k + \hat{\mathbf{d}}. \end{cases}$$

where $\mathbf{d}_k$ is the displacement estimate at iteration k and $\hat{\mathbf{d}}$ is the solution of

$$\mathbf{G}\hat{\mathbf{d}} = \sum_{W} \Big[(I(\mathbf{m}, t) - I(\mathbf{m} + \mathbf{d}_k, t+1)) \, \nabla I(\mathbf{m}, t) \Big].$$

In this framework, a feature can be tracked reliably if a numerically stable solution to Eq. (4) can be found, which requires that **G** is well-conditioned and its entries are well above the noise level. In practice, since the larger eigenvalue is bounded by the maximum allowable pixel value, the requirement is that the smaller eigenvalue is sufficiently large. Calling λ_1 and λ_2 the eigenvalues of **G**, we accept the corresponding feature if $\min(\lambda_1, \lambda_2) > \lambda$, where λ is a user-defined threshold [ST94].

5 Motion estimation and sequence registration

A feature-based motion estimation technique consists of three main steps, performed on adjacent images: *extraction of features* from the images, *matching of features* and *estimation of the transformation* between the images.

5.1 Feature based 2D motion estimation

We have performed feature extraction and point correspondence with the feature tracker described in Section 4.
The tracker has three main features:

- Extraction of features, that is used on the first frame of the sequence
- Tracking of features from one frame to the following
- Re-extraction of the features, if necessary. This means that if the scene has changed there could be too few features to track. Indeed, if in theory to obtain an homography 4 features are enough, in practice is better to have much more than that, because the quality of the features is not always good. The re-extraction function will add new features to the ones still present in the current image.

Assuming that the parallax effect is negligible, the homography approximating the transformation between couple of images is easy to calculate.
The tracker produces a list of features coordinates for each image. For each couple of images, after all the features lost by the tracker have been discarded, the homography can be produced, using the method described in Section 3.

5.2 2D Image registration

Global registration establishes a mapping between each frame and an arbitrary reference frame. We have described a mapping between an image and another through a homography. Once we know how to calculate the homography between couple of images, a reference frame must be chosen, in order to warp each image of the sequence into the common reference frame:

- **Frame to fixed frame registration:** if the scene does not change too much, that is, if the overlapping between an arbitrary couple of images is not too small, a fixed reference image can be chosen and all the homographies between each image and the fixed one can be computed. At this point the homographies can be used to warp each image in order to fit the content of the reference one.

- **Adjacent frames registration:** if the sequence spans a wide area the tracking of the features will be more robust between contiguous images. To produce the global alignment, since the homography, as defined in (1) is a linear operator, the transformation between non-contiguous frames can be obtained by multiplying the transformation matrices of the in-between image frames. The transformation between the image I_i and the image I_j, where $i < j$, is

$$H_{i,j} = \prod_{k=i}^{j-1} H_{k,k+1} \tag{20}$$

In most of the cases that we present, the images are registered with respect to the first frame of the sequence. Once the global alignment have been completed, if we imagine to pierce all the aligned frames with a temporal line, we will intersect pixels that, in absence of parallax, correspond to the same world point.

5.3 Parallax-based 3D motion estimation and registration

As described in section 2 a homography relates two views when the camera motion is rotational or when the entire scene can be approximated by a single parametric surface, typically a plane. In practice a $2D$ alignment is a good *approximation* if those conditions are violated but the violations are small, for instance if the camera translates slowly or if the relative depth of the scene (ΔZ) is small compared to the distance between the camera and the scene (Z). In those cases the residuals will not be zero but they will be small and the mosaic construction has to be changed in order to reflect the parallax effects [SN96, IAB$^+$96, SA96] The plane+parallax representation of image motion provides a mean to register pieces of a scene with arbitrary depth[SA96]. This approach is based on the fact that the relative affine structure is invariant on the choice of the second view. This property has been used to solve a variety of $3D$ geometry from multiple views. If the regular affine structure is calculated between two views, then a third view can be related to the previous reference view, by specifying the new viewing parameters (the plane homography and the epipole):

1. given two views, ψ_i and ψ_i' in full correspondence ($\mathbf{m}_i \leftrightarrow \mathbf{m}_i', \;\; i = 0 \ldots n$);
2. recover the epipoles (8-point algorithm [Har95] or variations);
3. given 3 arbitrary points and the epipoles, compute the plane homography $\mathbf{H}_\Pi$;
4. scale $\mathbf{H}_\Pi$to satisfy $\tilde{\mathbf{m}}_0' \simeq \mathbf{H}_\Pi \tilde{\mathbf{m}}_0 + \mathbf{e}'$;
5. solve for γ_i (relative affine structure) in $\tilde{\mathbf{m}}_i' \simeq \mathbf{H}_\Pi \tilde{\mathbf{m}}_i + \mathbf{e}'\gamma_i$;
6. a new view ψ_i'' is represented by a new epipole $\mathbf{e}''$ and a new plane homography $\mathbf{H}_\Sigma$;
7. points in the third view satisfy $\tilde{\mathbf{m}}_i'' \simeq \mathbf{H}_\Sigma \tilde{\mathbf{m}}_i + \mathbf{e}''\gamma_i$

$\mathbf{e}''$ and $\mathbf{H}_\Sigma$ can be computed from 6 corresponding points between first and third view.

6 Mosaicing

Video mosaicing has recently attracted a growing interest in the field of digital video processing and analysis, in applications such as automatic indexing of video data (see [BMM99] for a recent review), video coding and video editing.

A mosaic is an image constructed from all the frames of a scene sequence, that gives a panoramic view of the scene. Mosaics are a useful way to represent the information contained in video sequences. Since the images belonging to a sequence usually have large overlapping, a mosaic of the sequence provides a significant reduction in terms of space.
There are a lot of possible description of a scene that can be chosen depending on the scene in exam. According to Anandan et al. [IAB+96]:

- **Salient still [MB96, MP94]**: Static mosaics have been previously referred as *salient stills* of simply mosaics. They are usually built batch mode, by aligning all frames of a sequence to a reference coordinate system, which can be either user-defined or chosen automatically according to some criteria and by integrating all the images in a single mosaic image. The only information that are difficult to capture are the changes in the scene with respect to the background. Once a suitable way of dealing with moving objects has been found, static mosaics, efficient scene representation, ideal for video storage and retrieval, can also be successfully used for image stabilization, video compression, content-based layered representation of information.

- **Dynamic mosaic**: The only real limit of static mosaic is that they often must be constructed batch mode, for this reason they cannot completely follow the dynamic aspect of a video sequence. For this reason in various situations dynamic mosaics have been chosen. The content of a dynamic mosaic is variable and is constantly updated with the information of the current frame. When the first frame is read, the mosaic will coincide with the frame itself. In the further steps, the mosaic will be updated in order to be coherent with the latest frame read [IAB+96, SA96].

- **Multiresolution mosaic**: Changes in image resolution can occur within a sequence if the camera zooms in or out. If the mosaic is built at a low resolution, it will contain less information that the one that would have been available in the original sequence. On the other hand building the mosaic at the higher detected resolution, can cause oversampling of the low resolution frames. This problem can be handled with a *multi-resolution* structure with captures information from each new frame at its highest resolution level. In this way all the possible information is stored.

6.1 Sequence alignment

In this section we deal with the problem of creating a mosaic from a sequence of images. The construction of a mosaic is accomplished in three stages: *motion estimation*, *registration* and *rendering*. Motion estimation and registration have been described in general in the previous sections.
In this case the $2D$ motion estimation and alignment of the image frames of the sequence can be performed in three ways [IAB+96]:

- **Adjacent frames:** the homographies are computed between successive frames of the sequence. They can be composed to obtain the alignment between *any* two frames of the sequence.

- **Frame to mosaic:** to limit the problem of misalignments, for every new frame a temporary mosaic can be built and the new homography is computed between it and the new frame. This approach is alternative to the one of global alignment and further blending.

- **Mosaic to frame**: if one wants to maintain each image in its coordinate system it can be better to align the mosaic to the current frame.

If one needs to use a parallax-based $3D$ model, given a sequence of images with full correspondences between adjacent ones, one can compute, for each view, the plane homography and the relative affine structure, using the previous view as the "second view", and then warp it to a reference view (the "third" view) using the appropriate view parameters, that is the new correspondences between the current image and the reference one.

6.2 Mosaic rendering

Once the images have been aligned, they can be integrated, using a temporal filter, into a mosaic image. Several temporal filters can be used to construct the mosaic image. They all work on the intensity values belonging to the temporal line of each pixel.

- The *temporal average* of the intensity values. Moving objects would leave a "ghost-like" trace into the mosaic. It is effective in removing temporal noise.
- The *most recent information* that is, the entire content of the most recent frame is used to update the mosaic. A variation of this is used in general in the dynamic construction described above.
- The *temporal median* of the intensity values. Moving objects whose intensity patterns are stationary for less than half of the frames, tend to disappear in the resulting mosaic. In practice, moving objects are treated as outliers. The results are sharper than the ones obtained with temporal average.
- *Weighted temporal average* or *weighted temporal median* where the weights decrease with the distance of the pixel from the frame center. This scheme aims at ignoring distortions in the original sequence.

Other temporal filters have been presented in literature. For a wide panoramic see [IAB+96].

7 Stabilization

Image stabilization consists compensating for the camera motion by applying a suitable transformation to the image. In the stabilized image, scene points are motionless in spite of camera motion.
Image stabilization is really close to mosaic construction, indeed from the stabilized sequence a mosaic is obtained by merging the frames, that are already registered.
In order to compensate for the relative motion of the camera, we need to compute the homographies that map each frame onto a given *reference image*. In the warped images, static scene points are (ideally) motionless. We assume that each frame in the sequence overlaps with the reference one. There is no point in stabilizing an image which does not overlap with the reference one; in this case the latter should be changed.
Usually, the global registration is performed using the first image frame as a reference and all the other frames are mapped onto it, either directly with a frame to fixed frame motion estimation or with a adjacent frames motion estimation followed by a global alignment.

8 Segmentation of moving objects

In this section is described a method to segment moving objects in image sequences using a mosaic-based technique.
We first describe what changes in presence of moving object and how the results obtained can be used to perform motion segmentation.
After constructing the mosaic with simple features based registration, moving objects are segmented out by computing the grey-level differences between the stable background (mosaic) and the current frame. Similar approaches to ours have been used in the field of surveillance and targeting, where the ego motion of the camera is compensated before extracting moving targets from the background. In [CM99, SA96] the motion is computed for every pixel with a robust technique, and outliers masks give the moving object. In [GJ98] temporal analysis of gray levels, based on probabilistic models and a-priori information, is carried out in order to segment moving objects.
The motion estimation must be performed with a robust technique as the one described in section (3.1). This will produce an estimation of the motion of the background, that is the relative motion of the camera w.r.t. the scene, provided that the object moving inside the scene is not too big. At this point a sequence registration can be performed in the usual way. To segment out the moving object, or equivalently to build the background, a suitable temporal filter must be chosen, for instance the median or the weighted median. The blending stage will produce a mosaic of all the background elements, while the object moving will be disappeared. It is important to notice that to use the median filter all the frames need to be registered in advance, since it does not exist an incremental optimal estimator of the median operator. With a mosaic to frame registration (that is a back registration of the mosaic onto every single frame of the image sequence) a synthetic sequence of the background without the moving object can be obtained.

The foreground can be obtained with change detection and motion segmentation techniques that are now well known in literature [HNR84, Hua81, IRP94], comparing each frame of the virtual sequence with the corresponding frame of the original one.
Irani et al. [IAB$^+$96] suggest to choose the same *local misalignment measure*, S_t, that they successfully used in their motion analysis works:

$$S_t(x,y) = \frac{\sum_{(x_i,y_i)\in N(x,y)} |I_t(x_i,y_i) - I_t^{pred}(x_i,y_i)|}{\sum_{(x_i,y_i)\in N(x,y)} |\bigtriangledown I_t(x_i,y_i)| + C}. \tag{21}$$

Their method is based on a temporal continuity between the frames compared, that are, in our case, the mosaic and the current frame. Indeed, if the mosaic is built dynamically[IAH95], it is consistent with the most recent frame i.e., the mosaic contains all the information of the latest frame, for instance all the moving objects present in the latest frames. There exists, therefore, a temporal coherence between mosaic and current frame. Instead, if the mosaic is still, this coherence does not exist any more, since a possible moving object in the mosaic can be blurred or, as in our case, it has been removed. There is, thus, a strong spatio-temporal discontinuity between mosaic and frame that decrease the significance of the misalignment measure in itself. A difference-based technique seems to be more effective to our purposes. A grey level difference is performed between each original frame and the equivalent virtual one. Finally the result is thresholded to obtain a binary map.

The binary motion map contains the blobs produced by the moving objects and other smaller blobs due to misalignments, change in illumination or noise.
To segment out only the objects in motion, a simplifying assumption has been made: we assumed that only one object was moving in the scene. A generalization is currently under investigation. We detected the object in the first frame by choosing the area of the binary map containing the bigger connected component of moving pixels. After this initialization, on every frame i, the centroid of the largest connected component of its binary map is computed. The connected component of the $i+1$-th binary map chosen is the closest to the previous centroid.

At this point post-processing of the resulting maps is also needed, in order to obtain good quality segmentations. The morphological operator of *closure* [Ser82], that is *dilation* and *erosion* in cascade, produce a more compact blob, without adding noise and without altering its original dimension.

8.1 Description of the algorithm

Constructing the mosaic:

```
(1) for each new image of the sequence I_i
        calculate the homography wrt the previous one
        perform global registration wrt I_ref: calculate H(ref,i)

(2) build background mosaic blending all the images with a median filter
```

Object segmentation:

```
(1) for each  image of the sequence I_i
        warp the mosaic onto the image ref frame I_ref
        binary map B_i with a thresholded difference between I_i and I_ref

(2) B_0: find the main connected component
         compute the centroid c_0
    for each binary map B_i:
        find the connected component closest to c_(i-1)
        compute the centroid c_i

(3) for each B_i
        morphological post-processing of B_i
        build a grey level map (foreground) merging B_i and I_i
```

9 MPEG4 video coding

In this section we describe how the segmentation method explained above can be used to perform MPEG4 video compression.
The last MPEG standard, MPEG-4 [KPC97], follows an approach that is called *content-based* [WA94, KPC97], based on the way of perceiving a scene typical of the human brain. In a content-based approach the coding algorithm must describe the semantical meaning of the different objects. MPEG4 relies on a segmented representation of the video data, in order to achieve content-based

manipulation of image sequences. A scene is considered to be composed of several Video Objects (VOs). Each VO is characterized by intrinsic properties such as shape, texture, and motion. In this context, the term object has a very general interpretation, and it is not necessarily a physical object. For example, the background region may be considered as one VO.
A *sprite* consists of those regions of a VO that are present in the scene, throughout the video segment. An obvious example is the 'background sprite', which would consist of the mosaic of the background in a camera-panning sequence.
The MPEG-4 standard does not prescribe the method for creating VOs, it simply provide a standard convention for describing them, such that all compliant decoders are able to extract VOs from the encoded bit-stream.
If we think of the mosaic background and the foreground sequence as VOs, the idea described in the previous section can be seen as an MPEG-4 compliant content-based encoding technique.
In order to give a sketch of a whole video coding system, let us describe the decoder functioning. The large panoramic mosaic of the background (a sprite, in MPEG-4 terminology) is transmitted to the receiver only once. The moving foreground object is transmitted separately as an arbitrary-shape VO, described in the mosaic reference frame. Finally all the transformations between mosaic and original sequence (that is the mosaic to frame transformation) are needed. Actually it will suffice to transmit all the homographies between consecutive frames, since, starting from them, we can obtain every transformation from one reference frame to the other (using Eq. (20)). In the decoding phase, to build the original sequence, all we have do is to map the mosaic onto the frame of each image and paste the foreground onto it. At this point the original sequence is rebuilt.
Content-based representation, allows editing operations on the sequence, like inserting novel Video Objects thereby creating a realistic synthetic sequences. An example of this technique will be described in Section 10.

10 Results

In this section we will show some results on several application of 2D image registration, namely mosaicing, stabilization, motion segmentation and video representation, content-based manipulation. The latter is not strictly an application of image registration, but it take advantage of the content-based representation achieved before.

10.1 Mosaic construction

Our experiments to still mosaics constructions, have been mainly focused on sequences acquired in the underwater environment. To build the mosaic we have chosen the frame to frame (or adjacent frames) approach. Figure (2) shows the frames 0, 60, 119 (last) of the sequence "Underwater1", while Fig. (3) shows the resulting mosaic obtained with a temporal filter that, at every step, upgrade the mosaic with the latest information available.
Figure (3) presents three frames of the sequence "Underwater2" (frames 0, 70, 138). The mosaic is shown in Fig. (5).
Other examples of still mosaics can be find in the next figures (from Figure (6) to Figure (12)). Figures (6)and (7) show other two examples of underwater mosaics.
Figures (8) and (9) another interesting application environment of mosaic techniques: art. They show sequence frames and the final mosaic of the "Battistero di Padova" cupola, a masterpiece of Giusto de' Tornabuoni (refs).

Figure (10) shows the resulting mosaic of an out-door scene of the facade of Mount Batten Building, in Heriot-Watt. This mosaic is interesting because the radial distortion ...
Figures (11), (12) and (13) show 3 different results obtained from the same sequence changing the temporal integration. The difference are due to a person in front that was moving.

Figure 2: First, central and last frame from the sequence Underwater1.

Figure 3: Mosaic of the sequence Underwater1.

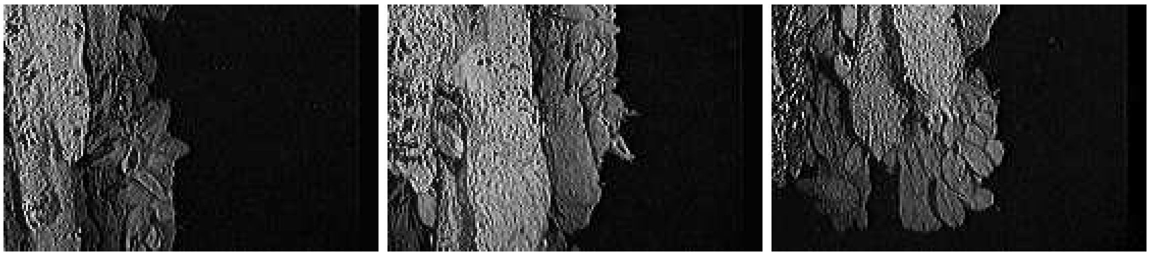

Figure 4: First, central and last frame from the sequence Underwater2.

10.2 Stabilization

Figure (14) shows the frames 0, 35, 45 and 89 of the sequence "Clio" made of 90 frames. The white box represents the position of the reference frame (in our case frame 0) with respect to the current one. The small white cross at the centre of the box represents a reference point that allows to see how the same pixel in different frames of the stabilized sequence, has the same content.

Figure 5: Mosaic of the sequence Underwater2.

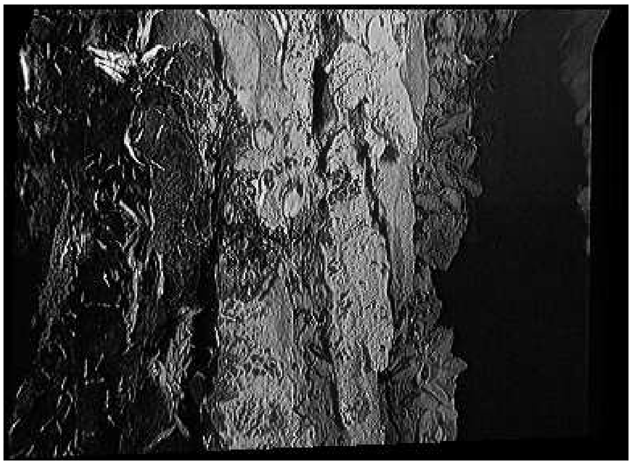

Figure 6: Mosaic of the sequence Rock1.

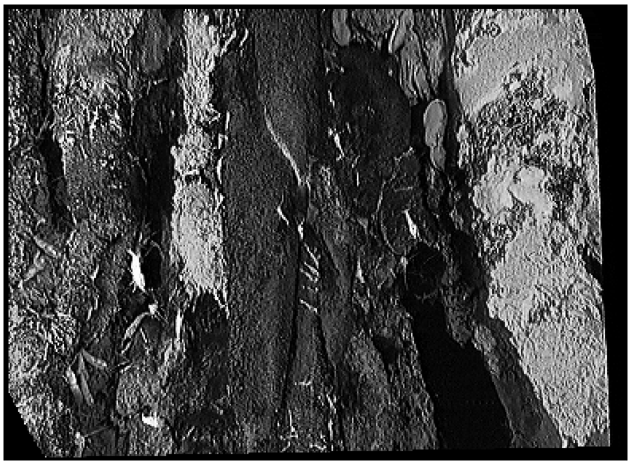

Figure 7: Mosaic of the sequence Rock2.

Figure 8: First, central and last frame from the sequence Affresco.

Figure 9: Mosaic of the sequence Affresco.

Figure 10: Mosaic of the sequence Facade. Notice the effects of radial distortion.

Figure 11: Mosaic of the sequence g78-3. An average filter has been used to blend the final mosaic

Figure 12: Mosaic of the sequence g78-3. The first image has been put into the mosaic, and in the next steps only the new pixels have been added

Figure 13: Mosaic of the sequence g78-3. Every step all the information of the latest image is put onto the mosaic

Since stabilization and mosaic building are closely related, to construct the mosaic from the stabilized sequence is straightforward (Figure 15).

10.3 Segmentation of moving objects and video coding

Figure (16) and (17) show frames of the two sequences that we use to describe our results in segmenting moving objects. They have been acquired with a hand-held camera. The first sequence is an outdoor scene with a car driving from the left to the right of the image field of view. The ego motion is nearly rotational, but a small translational component is present. The second sequence has a slightly different nature, the object (person) in motion is bigger and the natural environment under the sun produces a lot of shadows. The depth of the scene changes from the beginning of the sequence to the end. In spite of the fact that the camera motions are not exactly rotational and the scenes are not planar, the results obtained are quite satisfactory.
Figures (18) and (19) show the mosaics of the backgrounds obtained with the method explained in Section 5.
The background is mapped onto the original sequence frames and the residual analysis is performed. Figure (20) (left) shows the results obtained by using a thresholded difference between the 28-th frame of the sequence "Manuel" and its background. In Figure (20) (right) the results obtained with a gradient based segmentation are shown. As we explained in Section 8, we reckon that differences are more suitable to our purposes.
Figure (21) illustrates results of final segmentation, while Figure (22) contains a frame that have been encoded and decoded and the differences between the same frame and the original one.
As a measure of compression quality we have chosen the *point signal to noise ratio* (PSNR) on the difference of each original image of the sequence with the corresponding coded-decoded one. Given a source image f, $n \times m$, and a reconstructed image F, obtained by decoding the encoded version of f,

$$PSNR(f,F) = 20 \;\; log_{10} \; \frac{255}{MeanSquareError(f,F)} \tag{22}$$

where MSE, the mean square error, is

$$MSE = \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} (f(i,j) - F(i,j))^{1/2}. \tag{23}$$

The graphics in Figure (23) show that the quality of the compression does not degrade too much throughout the sequence.

Figure 14: Frames (0, 35, 45, 89) stabilized from the sequence Clio. The first one is the reference frame.

Figure 15: The mosaic of the sequence Clio

Figure 16: Frames 0, 50, 99 (the last) from "Manuel" sequence.

Figure 17: Frames 0, 20, 40 (the last) from "Super5" sequence.

Figure 18: Mosaic of Manuel (background sprite).

Figure 19: Mosaic of Super5 (background sprite.)

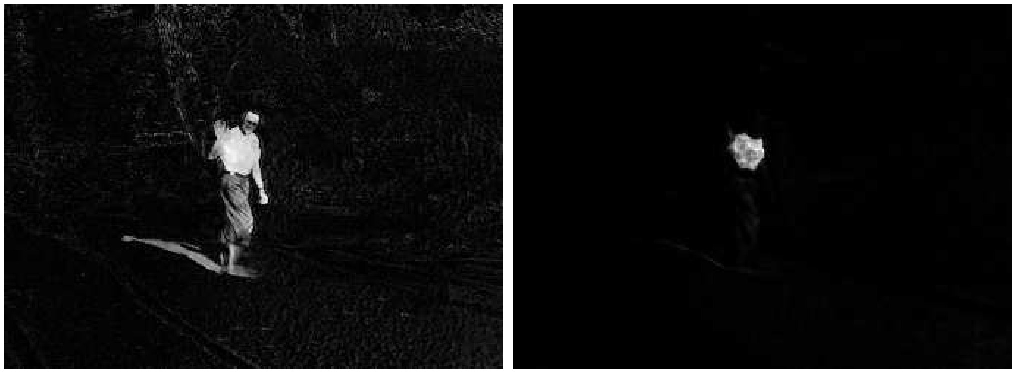

Figure 20: Residual analysis with differences (left) and normal flow (right).

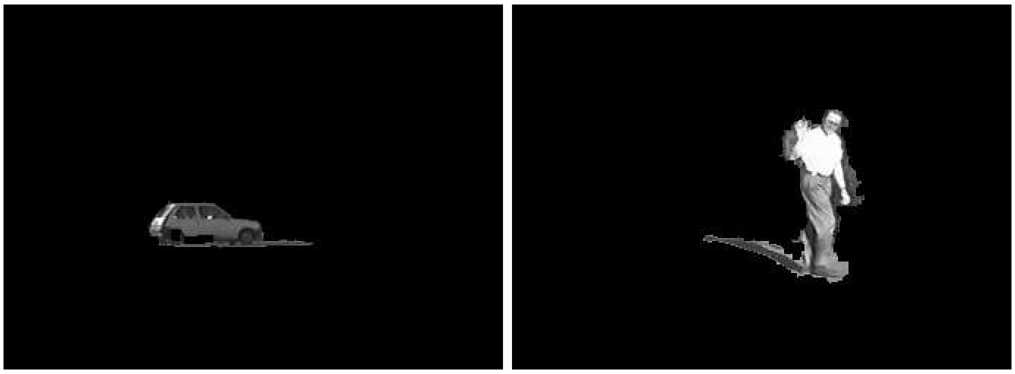

Figure 21: Moving objects extracted from the sequences.

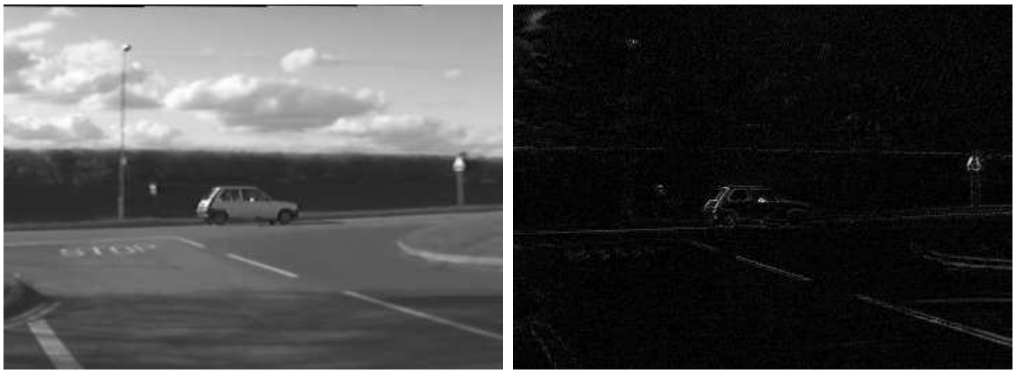

Figure 22: Example of a frame from encoded/decoded Super5 and differences with the original one (right).

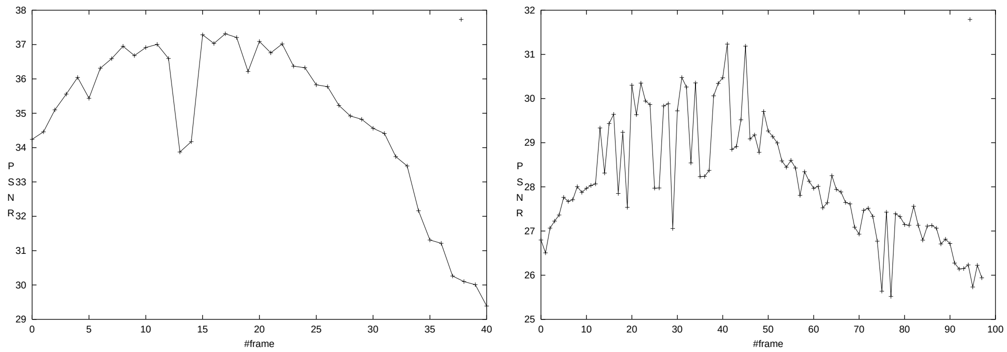

Figure 23: Power signal to noise ratio

Figure 24: Metrically rectified mosaic and a sample frame of the synthetic advertisement sequence.

10.4 Content-based manipulation

In this section we will describe an example of content-based manipulation of a video sequence, in which the segmented representation is exploited to insert a synthetic objects into the background, such as an advertising poster. The idea is to edit the background mosaic, then use the same decoding procedure as described in Section 9 to to create a new realistic sequence. The insertion of the synthetic sign is done on the *metrically rectified* mosaic. After editing, the rectified mosaic is then is warped back onto its original plane, hence the synthetic sign gets slanted accordingly. Figure (24) presents a result of video editing. On the left there is the metrically rectified mosaic of the background. All the editing operations can be performed on this image. At this point a back transformation brings the modified background in its original reference frame, so that the altered sequence can be built correctly (Figure (24), right).

More examples and sequences are available at the web page:
http://www.cee.hw.ac.uk/~fusiello/mosaic_demo/.

11 Conclusions and open issues

This report accounts for our research activity in the applications of 2D image registration.
After a careful exploration of the literature, we concentrated on some aspects of particular interest both for us and for our host group. In particular we developed sound code for homography computation, image mosaicing, image stabilization, motion segmentation, video coding/decoding, which our colleagues might find useful. Yet, there are still a few open issues.
Our work could be extended in several directions. As for the mosaics, multiresultion, cylindrical (or spherical) projection could be added without modifying the approach chosen. More structural changes would be required to accommodate for long sequences by employing sub-mosaicing, or to perform on line construction of the mosaic by finding a recursive estimator which could substitute the median. A big issue in mosaicing is the objective evaluation of the quality of the result. No attempt have been made in the past, to the best of our knowledge. The use of image quality indices such as Tenengrad, used for estimating the defocusing, or the characterization of the quality in terms of the Fourier spectrum could be investigated.
As for the motion segmentation, structural improvements would be the introduction of an automatic procedure for computing the threshold used on the difference images and the extension of the method to multiple moving objects, appearing and disappearing in any frame.

Our current method can be seen as a very simple target tracker based on the assumption that only one object is moving and that it appears in the first frame. Without modifying the motion detection module, a more complex tracker can be plugged-in, which caters for multiple targets in a clutter [BSF88].

acknowledgements

We wish to thank Emanuele Trucco and Francesco Isgrò for all the useful discussions and the suggestions they gave during the course of this work. Thanks are also due to all the people of the Ocean Systems Laboratory, Heriot-Watt University, and in particular to Costas Plakas that implemented the Feature Tracker code. Andrea Fusiello has been supported by EPSR (Grant GR/M40844), Francesca Odone by a Marie Curie Training and Mobility Grant (ERB4001GT-97-3072).

A Two-view geometry

If we take the first camera reference frame as the world reference frame, we can write the two following general camera matrices:

$$\tilde{\mathbf{P}} = \mathbf{A}[\mathbf{I}|\mathbf{0}] = [\mathbf{A}|\mathbf{0}] \qquad (24) \qquad\qquad \tilde{\mathbf{P}}' = \mathbf{A}'[\mathbf{R}|\mathbf{t}] \qquad (25)$$

Let
$$\tilde{\mathbf{w}} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad \text{and} \quad \tilde{\mathbf{m}} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix},$$

be the 3D point and its projection onto the first camera, respectively, Then

$$\kappa \tilde{\mathbf{m}} = \tilde{\mathbf{P}} \tilde{\mathbf{w}}, \tag{26}$$

where κ is the depth of $\mathbf{w}$, that is, its distance from the focal plane of the first camera. Similarly, for the second camera we can write:

$$\kappa' \tilde{\mathbf{m}}' = \tilde{\mathbf{P}}' \tilde{\mathbf{w}}. \tag{27}$$

From (26) and (25) we obtain:

$$\kappa' \tilde{\mathbf{m}}' = \mathbf{A}'[\mathbf{R}|\mathbf{t}] \tilde{\mathbf{w}} = \mathbf{A}'[\mathbf{R}|\mathbf{t}] \left(\begin{bmatrix} x \\ y \\ z \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \right) = \mathbf{A}'\mathbf{R} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \mathbf{A}'\mathbf{t}, \tag{28}$$

and from (27) and (24) we obtain:

$$\kappa \mathbf{A}^{-1} \tilde{\mathbf{m}} = [\mathbf{I}|\mathbf{0}]\ \tilde{\mathbf{w}} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}. \tag{29}$$

Substituting the latter in (28) yields

$$\kappa' \tilde{\mathbf{m}}' = \kappa \mathbf{A}'\mathbf{R}\mathbf{A}^{-1} \tilde{\mathbf{m}} + \mathbf{A}'\mathbf{t}. \tag{30}$$

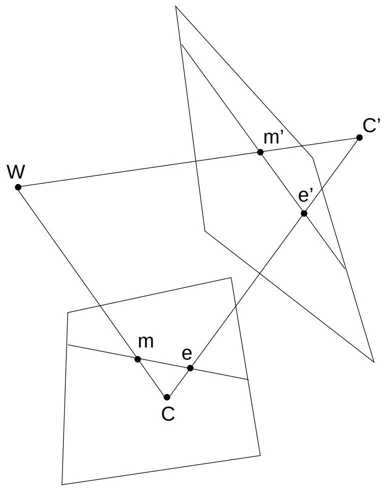

Figure 25: The epipolar geometry

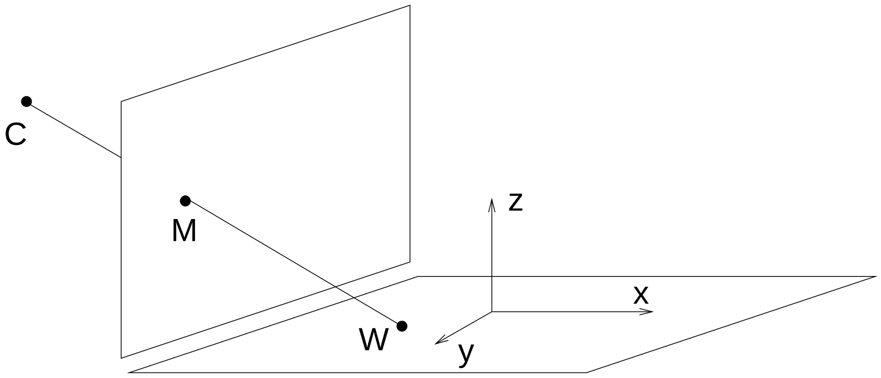

Figure 26: The map between a world plane and a perspective image is an homography

B Metric rectification

The map between a 3D point $\mathbf{w}$ and its projection onto the image plane is given by a 3×4 matrix $\mathbf{P}$ (in homogeneous coordinates) such that:

$$\kappa \tilde{\mathbf{m}} = \tilde{\mathbf{P}} \tilde{\mathbf{w}}, \tag{31}$$

where κ is the depth of $\mathbf{w}$.

The map between a world plane and a perspective image is a homography (a plane projective transformation). The easiest way to see it, is to choose the world coordinate system such that the plane of the points have zero z coordinate.
Then the perspective projection matrix $\tilde{\mathbf{P}}$ reduces to

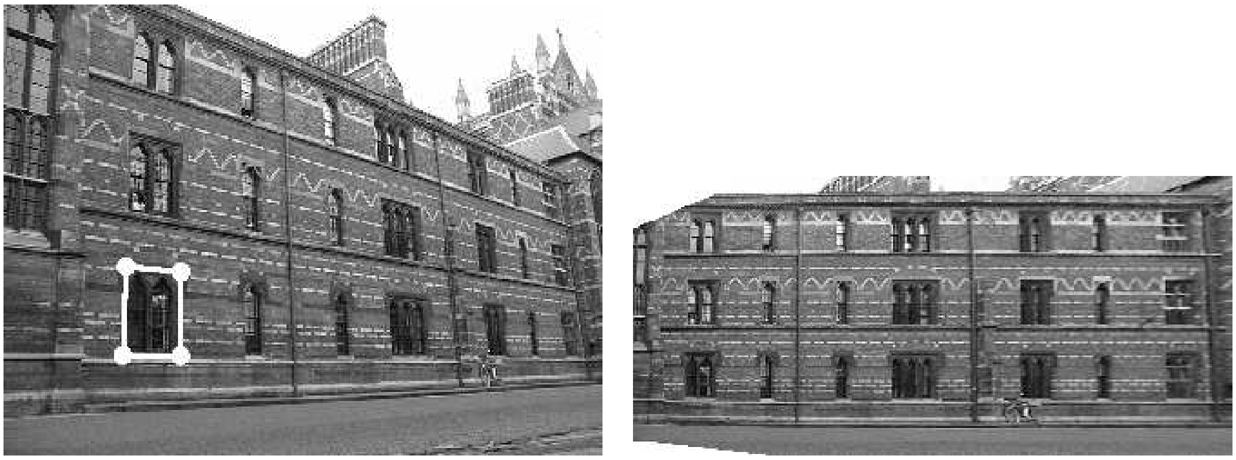

Figure 27: A perspective image and a metric rectified image of the plane (from [LCZ99]).

$$\kappa \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} P_{1,1} & P_{1,2} & P_{1,3} & P_{1,4} \\ P_{2,1} & P_{2,2} & P_{2,3} & P_{2,4} \\ P_{3,1} & P_{3,2} & P_{3,3} & P_{3,4} \end{bmatrix} \begin{bmatrix} x \\ y \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} P_{1,1} & P_{1,2} & P_{1,4} \\ P_{2,1} & P_{2,2} & P_{2,4} \\ P_{3,1} & P_{3,2} & P_{3,4} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{32}$$

which is a 3×3 matrix representing a general plane to plane projective transformation.
A homography is fully defined by four points of which we know the relative position in the world plane. Once the homography is determined, the image can be back projected onto the object plane. This is equivalent to synthesize an image from a fronto-parallel view of the plane. This is known as metric rectification [LZ98] of a perspective image.
If four points cannot be measured a *stratified* approach is used. The projective transformation **H** can be decomposed into a concatenations of three matrices **M**, **C** and **V** representing similarity, affine and pure projective transformations respectively:

$$H = MCV \tag{33}$$

- the similarity matrix **M** (four parameters) can be ignored;
- geometry is recovered up to an affine transformation by applying projective matrix **V**:

$$V = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ l_1 & l_2 & l_3 \end{bmatrix} \tag{34}$$

  where $l_\infty = (l_1, l_2, l_3)$ is the vanishing line that can be recovered from two vanishing points, or a set of equally spaced lines on the plane;
- recovery of metric geometry requires the affine matrix **C**:

$$C = \begin{bmatrix} 1/\beta & -\alpha/\beta & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{35}$$

The two parameters α and β can be computed from the intrinsic parameters [LCZ99] or from at least two independent constraints chosen among the following:

- a known angle
- equality of two (unknown) angles
- a known length ratio

C Computing things from homographies

Homography $\mathbf{H}_\Pi$ induced by a plane, known intrinsic parameters

Using the definition of $\mathbf{H}_\infty$ we get

$$\mathbf{H}_\Pi = \mathbf{A}'(\mathbf{R} + \mathbf{t}\frac{\mathbf{n}^\top}{d})\mathbf{A}^{-1}. \tag{36}$$

Therefore motion parameters $\mathbf{R}$ and $\mathbf{t}$ as well as plane parameters $\mathbf{n}$ and d are encoded by $\mathbf{H}_\Pi$.

$$\mathbf{A}'^{-1}\mathbf{H}_\Pi\mathbf{A} = d\mathbf{R} + \mathbf{t}\mathbf{n}^\top. \tag{37}$$

By taking the SVD factorization of $\hat{\mathbf{H}}_\Pi = \mathbf{A}'^{-1}\mathbf{H}_\Pi\mathbf{A}$ we get

$$\hat{\mathbf{H}}_\Pi = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^\top \tag{38}$$

Hence

$$\mathbf{\Lambda} = \frac{d}{s}(s\mathbf{U}^\top\mathbf{R}\mathbf{V}) + (\mathbf{U}^\top\mathbf{t})(\mathbf{V}^\top\mathbf{n})^\top = d'\mathbf{R}' + \mathbf{t}'\mathbf{n}'^\top \tag{39}$$

where $s = \det(\mathbf{U})\det(\mathbf{V})$ (to ensure that $\mathbf{R}'$ is a rotation).
This equation can be solved (with a twofold ambiguity in the solution) to obtain: (i) the normal to the plane $\mathbf{n}$, (ii) the rotation $\mathbf{R}$ and (iii) the translation scaled with the distance to the plane ($\mathbf{t}/d$) [FL88].

Homography $\mathbf{H}_\infty$ of infinity plane, unknown but constant intrinsics

Intrinsic parameters can be computed (solving a linear system) from

$$\mathbf{H}_\infty\mathbf{A}\mathbf{A}^\top\mathbf{H}_\infty^\top = \mathbf{A}\mathbf{A}^\top. \tag{40}$$

Then the rotation can be extracted from $\mathbf{H}_\infty = \mathbf{A}'\mathbf{R}\mathbf{A}^{-1}$.
This is a very simple *autocalibration* technique[Har97], that has been extended also to varying intrinsics[dHH99].
$\mathbf{H}_\infty$ can be obtained

- by knowing at least 3 vanishing point;
- by approximating it with a plane "far enough" from the camera[VZR96];
- by causing the camera to rotate (more frequently)[Har97, dHH99].

References

[BFB94] J. L. Barron, D. J. Fleet, and S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, 1994.

[BMM99] R. Brunelli, O. Mich, and C. M. Modena. A survey on the automatic indexing of video data. *Journal of Visual Communication and Image Representation*, 10:78–112, 1999.

[Bro92] L. G. Brown. A survey on image registration techniques. *ACM Computing Surveys*, 24(4):325–376, December 1992.

[BSF88] Y. Bar-Shalom and T. E. Fortmann. *Tracking and data Association.* Academic Press, 1988.

[CM99] I. Cohen and G. Medioni. Detecting and tracking moving objects in video surveillance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages II:319–325, 1999.

[CV92] M. Campani and A. Verri. Motion analysis from first order properties of optical flow. *CVGIP: Image Understanding*, 56:90–107, 1992.

[CZ98] David Capel and Andrew Zisserman. Automated mosaicing with super-resolution zoom. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 885–891, 1998.

[Dav98] James Davis. Mosaics of scenes with moving objects. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 354–360, 1998.

[dHH99] L. deAgapito, R.I. Hartley, and E. Hayman. Linear calibration of a rotating and zooming camera. In *CVPR99*, pages I:15–21, 1999.

[FL88] O.D. Faugeras and F. Lustman. Motion and structure from motion in a piecewise planar environment. *International Journal of Pattern Recognition and Artificial Intelligence*, 2:485–508, 1988.

[GJ98] P.R. Giaccone and G.A. Jones. Segmentation of global motion using temporal probabilistic classification. In *British Machine Vision Conference*, pages 619–628, 1998.

[GL96] G. H. Golub and C. F. Van Loan. *Matrix Computations.* The John Hopkins University Press, third edition, 1996.

[GM99] A. A. Goshtasby and J. Le Moigne, editors. *Pattern Recognition. Special Issue: Image Registration*, volume 32. Pergamon, January 1999.

[GSV98] Nuno Gracias and Joseé Santos-Victor. Automatic mosaics creation of the ocean floor. In *Proceedings of the OCEANS Conference*, 1998.

[Har95] R. I. Hartley. In defence of the 8-point algorithm. In *Proceedings of the IEEE International Conference on Computer Vision*, 1995.

[Har97] Richard Hartley. Self-calibration of stationary cameras. *International Journal of Computer Vision*, 22(1):5–24, February 1997.

[HNR84] Y. Z. Hsu, H. H. Nagel, and G. Rekers. New likelihood test methods for change detection in image sequences. *Computer Vision, Graphics, and Image Processing*, 26:73–106, 1984.

[Hua81] T. S. Huang. *Image sequence analysis.* Springer-Verlang, Berlin-Heidelberg-New York, 1981.

[IAB+96] M. Irani, P. Anandan, J. Bergen, R. Kumar, and S Hsu. Efficient representations of video sequences and their applications. *Signal processing: Image Communication*, 8(4), 1996.

[IAH95] M. Irani, P. Anandan, and S. Hsu. Mosaic based representations of video sequences and their applications. In *International Conference on Computer Vision*, pages 605–611, 1995.

[IRP94] M. Irani, B. Rousso, and S. Peleg. Computing occluding and transparent motions. *International Journal of Computer Vision*, 12(1):5–16, February 1994.

[KPC97] R. Koenen, F. Pereira, and L. Chiariglione. MPEG-4: Context and objectives. *Signal Processing: Image Communications*, 9(4):295–, 1997.

[LCZ99] D. Liebowitz, A. Criminisi, and A. Zisserman. Creating architectural models from images. In *EUROGRAPHICS'99*, 1999.

[LZ98] D. Liebowitz and A. Zisserman. Metric rectification for perspective images of planes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 482–488, 1998.

[MB96] M. Massey and W. Bender. Salient stills: process and practice. *IBM Systems Journal*, 35(3,4), 1996.

[MC97] Carlos Morimoto and R. Chellappa. Fast 3D stabilization and mosaic construction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 660–665, 1997.

[MMKR91] P. Meer, D. Mintz, D. Y. Kim, and A. Rosenfeld. Robust regression methods in computer vision: a review. *International Journal of Computer Vision*, 6:59–70, 1991.

[MP94] S. Mann and R. W. Picard. Virtual bellows: Contructing high quality stills from video. In *IEEE International Conference on Image Processing*, 1994.

[RL87] P. J. Rousseeuw and A. M. Leroy. *Robust regression & outlier detection.* John Wiley & sons, 1987.

[RPFRA98] B. Rousso, S. Peleg, I. Finci, and A. Rav-Acha. Universal mosaicing using pipe projection. In *International Conference on Computer Visione (ICCV98)*, 1998.

[SA96] H. Sawhney and S. Ayer. Compact representations of videos through dominant and multiple motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):814–830, August 1996.

[Ser82] J. Serra. *Image Analysis and Mathematical Morphology.* Academic Press, 1982.

[SK52] J. G. Semple and G. T. Kneebone. *Algebraic projective geometry.* Oxford University Press, 1952.

[SK97] Harpreet S. Sawhney and Rakesh Kumar. True multi-image alignment and its application to mosaicing and lens distortion correction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 450–456, 1997.

[SN96] A. Shashua and N. Navab. Relative affine structure: Canonical model for 3D from 2D geometry and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9):873–883, September 1996.

[ST94] J. Shi and C. Tomasi. Good features to track. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, June 1994.

[Sze96] R. Szeliski. Video mosaics for virtual environments. *IEEE Computer Graphics and Applications*, 16(2):22–30, March 1996.

[TK91] C. Tomasi and T. Kanade. Detection and tracking of point features. Technical Report CMU-CS-91-132, Carnegie Mellon University, Pittsburg, PA, April 1991.

[TV98] Emanuele Trucco and Alessandro Verri. *Introductory Techniques for 3-D Computer Vision.* Prentice-Hall, 1998.

[VZR96] T. Viéville, C. Zeller, and L. Robert. Using collineations to compute motion and structure in an uncalibrated image sequence. *International Journal of Computer Vision*, 20(3):213–242, 1996.

[WA94] J. Y. A. Wang and E. H. Adelson. Representing moving images with layers. *IEEE Transactions on Image Processing*, 3(5):625–638, May 1994.

[ZFD97] I. Zoghlami, O. Faugeras, and R. Deriche. Using geometric corners to build a 2D mosaic from a set of images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 420–425, 1997.

[Zha97] Z. Zhang. Parameter estimation techniques: a tutorial with application to conic fitting. *Image and Vision Computing*, 15(1):59–76, 1997.