Wide-baseline Multiple-view Correspondences *

Vittorio Ferrari[1], Tinne Tuytelaars[2], Luc Van Gool[1,2]
[1]Computer Vision Group (BIWI), ETH Zuerich, Switzerland
[2]ESAT-PSI, University of Leuven, Belgium
{ferrari,vangool}@vision.ee.ethz.ch, Tinne.Tuytelaars@esat.kuleuven.ac.be

Abstract

We present a novel approach for establishing multiple-view feature correspondences along an unordered set of images taken from substantially different viewpoints. While recently several wide-baseline stereo (WBS) algorithms have appeared, the N-view case is largely unexplored. In this paper, an established WBS algorithm is used to extract and match features in pairs of views. The pairwise matches are first integrated into disjoint feature tracks, each representing a single physical surface patch in several views. By exploiting the interplay between the tracks, they are extended over more views, while unrelated image features are removed. Similarity and spatial relationships between the features are simultaneously used. The output consists of many reliable and accurate feature tracks, strongly connecting the input views. Applications include 3D reconstruction and object recognition. The proposed approach is not restricted to the particular choice of features and matching criteria. It can extend any method that provides feature correspondences between pairs of images.

1. Introduction

In the last few years, several wide-baseline stereo (WBS) matching algorithms, capable of finding correspondences between two images taken from substantially different viewpoints have appeared. In [1, 3, 4, 2] local features are extracted independently from the two images, then characterised by invariant descriptors and finally matched. The power of these approaches is twofold. First, local features bring tolerance to occlusions. Second, the construction process and descriptors are invariant under affine deformations, allowing large viewpoint changes, and under linear photometric transformations, allowing changes in illumination.

Tuytelaars and Van Gool [1] construct small image regions around corners (nearby edges provide orientation and skew, while scale and stretch are given by the extrema of a 2D affinely invariant function) and intensity extrema (based on the intensity profile along rays emanating from it). Matas *et al* [4] propose regions as connected components of pixels which are all brighter or darker than all pixels on the region's contour, or as regions bounded by cycles of edge pixels. Baumberg [3] describes regions with their location and scale determined by a multi-scale Harris corner detector, while an adaptive procedure based on the second moment gradient matrix recovers orientation and skew. Mikolajczyk and Schmid [2] present a similar extraction approach, but dealing with larger scale changes. Lowe [6] proposes the SIFT detector, where a feature's location and scale are determined by extrema of a DoG function in scale space, and its orientation by the dominant, local image gradient orientation. It is less immune against viewpoint changes, being only invariant under similarity transformations. Tell and Carlsson [7] offer an interesting alternative where a corner is characterized by intensity profiles on line segments connecting it to other corners. This avoids the problem of constructing affinely invariant regions, but it is less local and does not produce local shape descriptions.

While these methods have focused on two views, this paper presents a method for obtaining wide-baseline matches among a larger set of unordered images. The output is a set of *region-tracks*, each aggregating the image regions of a certain physical surface patch along several views. The only alternative approach to date is recently due to Schaffalisky and Zisserman [5]. Wide-baseline N-view matches are precious for a number of applications. In 3D reconstruction (e.g. from uncalibrated images), the 3D model of a scene could be generated from still images rather than video.In object recognition, appearance-based strategies could be developed with a limited number of reference views [6], bridging in fact the gap with model-based approaches.

In order to build region-tracks, a natural idea is to apply an established WBS algorithm to view pairs and then integrate the outputs. A naive implementation is bound to fail, due to two main problems. The first is structural: WBS algorithms return sets of matches between *pairs* of views. Region-tracks must be derived from this, even in the presence of matching errors, which can lead to contraddictory information about their composition. The second problem is statistical: suppose a physical region is visible in n views,

*This research was supported by EC project CIMWOS. Tinne Tuytelaars is a Postdoctoral Fellow of the Fund for Scientific Research Flanders (Belgium)

there is no guarantee that the image region will be extracted in all views. Suppose the probability p of extracting the region is the same, and independent, in every view where it is visible. The region will be extracted in its n views with probability p^n. Thus, even assuming a perfect matcher, the probability decreases exponentially with the number of views. Failures to match even lower the odds. We build upon a multi-scale extension of the WBS algorithm proposed in [1]. However, these problems are common to all WBS schemes based on independently extracted features. In practice, given 3 views v_1, v_2, v_3, the method typically finds many matches between view pair (v_1, v_2) and view pair (v_1, v_3), but the two sets of matches will often differ substantially. A smaller number of regions (typically half) are matched over the three views. It goes without saying that taking more views, the situation deteriorates further (e.g.: only a few, if any, 5-view matches can be found).

Without an *a priori* ordering of the input image set, we start by applying [1] to all pairs of views. This may seem computationally expensive, but the application is meant for wide baseline conditions, in which case there will only be a limited number of views (typically 10 to 30). Otherwise, other and more appropriate approaches than WBS could be applied. Besides, this allows to use all initial matches, which in turn reflects on the quality of the final results. In section 2 we propose an algorithm to construct clean, disjoint region-tracks from the large set of all pairwise matches, taking explicitly into account the inevitable presence of matching errors. The registration between the regions within each region-track is then improved by a novel way to refine the affine transformations between them (section 3). Unmatched regions are propagated to other views by exploiting the geometric and photometric transformation of nearby matches. This extends the region-tracks to previously uncovered views. This step is vital for escaping the aforementioned statistical trap. Finally, erroneous parts of the region-tracks, due to original mismatches and occasional mispropagations, are removed based on the analysis of the spatial arrangements of multiple regions in pairs of views (section 5). Section 6 presents experimental results and concludes the paper. The approach is not restricted to [1], but can complete any other WBS method based on local affine features.

2. Extracting region-tracks

The two-views matching algorithm [1] is applied between all pairs of views $(V_i, V_j), i \neq j$ producing separate sets of *pairwise* matches, in the form (A_l, B_m), indicating that region A in view V_l has been matched to region B in view V_m. In order to fulfill the main goal, we need to integrate this information into *region-tracks*: each aggregating the image-regions of a certain physical surface patch along the views. Such a region-track should be in the form $R = \{A_l, B_m, ..., J_z\}$, indicating that regions $A_l, B_m, ..., J_z$ are all in mutual correspondence, and represent a single physical region R as it's imaged on views $l, m, ..., z$. The region-tracks should be mutually disjoint, i.e.: no two region-tracks should share a common region.

Define a similarity measure between two regions A, B

$$\mathrm{sim}(A,B) = \mathrm{NCC}(A,B) + (1 - \frac{\mathrm{d}\overline{\mathrm{RGB}}(A,B)}{100}) \qquad (1)$$

where NCC is the normalized cross-correlation between the regions' greylevel patterns, $\mathrm{d}\overline{\mathrm{RGB}}$ is the average pixel-wise Euclidean distance in RGB color-space after independent normalization of the 3 colorbands (necessary to achieve photometric invariance). R, G, B ranges in $[0, 255]$. The two regions are aligned by the affine transformation mapping A to B, before computation. This mixed measure proved considerably more discriminant than NCC alone in various experiments.

How to get region-tracks out of pairwise matches ? Let's consider the graph G where vertices represent regions and edges are weighted by function (1). No edge is present between two unmatched regions. Suppose for a moment that the region extraction and two-views matching processes were perfect, so that there are no mismatches and no missing matches. In this ideal case, G will be composed of completely connected, disjoint subsets of vertices (*cliques*). Since each clique corresponds to a region-track, our task is easily solved (figure 1a).

Unfortunately real data is plagued by two kinds of errors: mismatches, which insert spurious edges in G, and missing-matches, which make G lack some correct edges (figure 1b). With these errors, G is no longer in a disjoint-cliques form and ambiguities about the composition of the region-tracks arise. Three properties come to help. First, matching is transitive: if (A_l, B_m) and (B_m, C_n) are matches then (A_l, C_n) *must* be a match as well. Second, a region cannot be matched to two different regions which are on the same view (one-to-one-constraint). Third, given the correct match (A_l, B_m) and the mismatch (A_l, C_m), the similarity measure is, almost always, higher for the former. We propose a conflict-resolution algorithm (CR) that exploits these properties to transform G into a disjoint-cliques form. CR reconstructs missing edges and discovers spurious edges by the same basic process of *triangulation*: given any two edges (A_l, B_m) and (B_m, C_n), the edge (A_l, C_n) is added to G if not yet present. The two edges generating the new one are its *parents*. An original edge has no parent (*leaf*). Two edges (A_l, B_m) and (A_l, C_m) are in *conflict*, as they violate the one-to-one-constraint. At least one edge in a conflict is caused by a mismatch. Adding an edge can generate a conflict, which is resolved by *removing* the least weighted edge (i.e.: the match with the lower similarity measure). When an edge is removed also the weakest

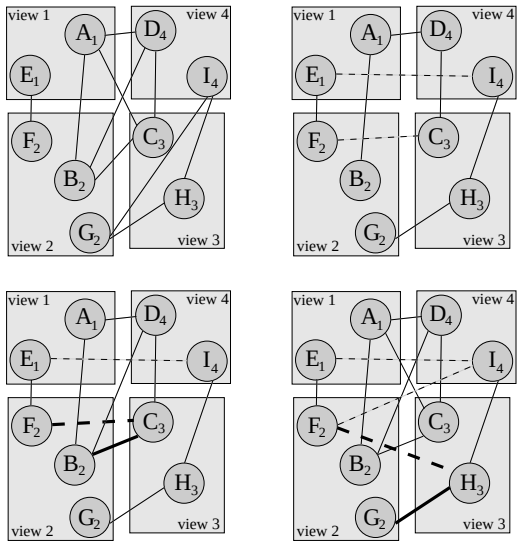

Figure 1: *a) ideal case, b) real case, dotted edges are mismatches, c) and d): two steps of CR. Thick edges are in conflict.*

of its parents is removed, unless the edge is a leaf. This is justified by the fact that, if an edge is considered a mismatch, then at least one of its parents *must* be regarded as a mismatch.

CR starts by triangulating the match with highest similarity. This new match is used to triangulate further. CR recurses until no new edge can be constructed with the latest generated one as a parent, or if this is removed as consequence of a conflict. At this point, the next match with highest similarity is triangulated, and so on. Cycles are avoided by not allowing the triangulation of a previously removed edge, thus ensuring CR's convergence. After termination, G is composed of disjoint cliques.

Figure 1b shows an interesting case. First, (B_2, D_4) is triangulated from (A_1, B_2) and (A_1, D_4). This in turn generates (B_2, C_3) (from (C_3, D_4) and (B_2, D_4)). Now (B_2, C_3) conflicts with (F_2, C_3), and the latter is removed as it has lower similarity measure (mismatch, figure 1c). Since (B_2, C_3) survived, it combines with (A_1, B_2) to give (A_1, C_3). No triangles can be further constructed based on (A_1, C_3), so original edges are processed. (F_2, I_4) is triangulated from (E_1, F_2) and (E_1, I_4), which then induces (F_2, H_3). A conflict between (F_2, H_3) and (G_2, H_3) is detected and causes the removal of (F_2, H_3) (figure 1d). This calls upon the removal of its weakest parent (F_2, I_4), which in turn causes the removal of grandparent (E_1, I_4). Finally, the last possible edge (G_2, I_4) is added and the algorithm terminates on the ideal solution (figure 1a).

This example shows how CR discovers and solves ambiguities in a conflictual set of matches. The conflicts are often hidden within the dataset, but are triggered by recursive triangulation, and then disambiguated by maximal similarity. CR is guaranteed to find all conflicts, although it's not sure to always solve them correctly: in a few cases the similarity of the mismatch might exceed the match's one. However, CR's goal is not to find the optimal solution, but instead to efficiently yield a reasonably good starting point for the further processing stages.

The proposed approach cleans the initial set of matches, yielding a coherent, conflict-free one, respecting the transitivity of matching and the one-to-one constraint. Hence region-tracks are extracted out of a large set of conflictual pairwise matches (1000s for 10 views). CR is time-efficient, as it evaluates the similarity measure only when needed, and cautious in that it has no fixed similarity threshold for rejecting a match, but instead only compares similarities, relying on the weaker assumption that a correct match scores higher than a wrong match. This is important in this initial phase where unnecessarily removing matches could compromise the performance of the further processing stages.

In the rest of the paper the following definitions are assumed. Γ is the set of all region-tracks. $R = \{R_v\}_{v \in vs}$ is a region-track, composed by image regions in views vs. If $R_v \in R$ we say that region-track R is *present* in view v. $\Phi_{vs} = \{R \in \Gamma | R_v \in R, \forall v \in vs\}$ is the set of region-tracks present simultaneously in each view $v \in vs$.

3. Refinement

Let R_1, R_2 be two regions matched between two views. In practical situations R_1 is not perfectly registered with R_2 as they are *independently* extracted from the two views. It is desirable to refine this initial match, so as to obtain a more accurate correspondence and to provide better input to the propagation stage (section 4).

Formally, R_2 should be affinely transformed such that the resulting region maximises the similarity (1) with R_1. This calls upon an algorithm that can browse a reasonably large range of the 6D affine space searching (an approximation of) the maxima while evaluating the similarity function as few times as possible (computational cost). The affine space is decomposed into its translation (t_x, t_y), scale (s_x, s_y), rotation (θ) and shear (h) components. We consider searching an *hypercube* Ω bounded in all its dimensions by predefined values [1]. A point in Ω is denoted $A = (t_x, t_y, s_x, s_y, \theta, h)$.

Let R_2^c be obtained by centering R_2 around $(0, 0)$. The algorithm starts from the identity transformation $A = A_0 = (0, 0, 1, 1, 0, 0)$ and searches for A_{max} so that

$$A_{max} = \arg\max_{A \in \Omega} \text{sim}(R_1, AR_2^c)$$

[1] In our experiments: $t_x \in [-14, 14], t_y \in [-14, 14], s_x \in [0.6, 1.8], s_y \in [0.6, 1.8], \theta \in [\frac{-Pi}{4}, \frac{Pi}{4}], h \in [-1, 1]$. Discretization: 2 for translations, 0.1 for scales, $\frac{Pi}{16}$ for rotation, 0.2 for shear

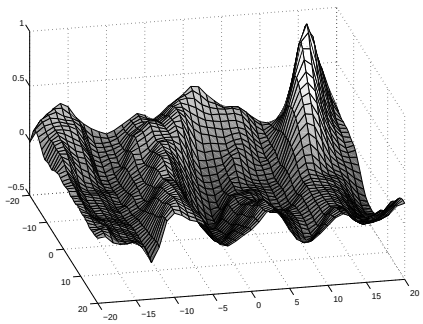

Figure 2: *Example 2-dimensional non-convex function to be maximized. Algorithms start from* $(0, 0)$*. Gradient Descent reaches point* $(3, 20)$*, while the proposed algorithm iterates through* $(0, -19), (15, -19)$ *and then reaches the global maximum* $(15, -15)$*. Notice the numerous foldings in the landscape.*

Let $A(dim, val)$ be point A with component dim set to val. Let i_d, f_d be the bounds of Ω along dimension d and

$$v^d_{max} = \max_{v \in [i_d, f_d]} \text{sim}(R_1, A(d, v)R^c_2) \quad (2)$$

be the value of the highest similarity induced by affine transformations along the straight line segment passing from A, starting at $A(d, i_d)$ and ending at $A(d, f_d)$. Computing (2) for all dimensions yields 6 values $v^d_{max}, d \in 1...6$, from which only the absolute maximum v_{best} is retained. This corresponds to evaluating similarity on 6 line segments concurrent in A (*rays*). Point A is now moved to the affine transformation inducing v_{best}. The process iterates until stability.

The proposed algorithm (RF) only searches a predefined, immutable hypercube Ω centered at A_0, and can be seen as a particular way of stepwise walking in Ω, where each step occurs in a direction parallel to a coordinate axis and can be arbitrary large.

RF is motivated by the observation that the similarity function generates spaces which are smooth, but highly non-convex, in which they show frequent and diverse foldings. In such a situation, Gradient Descent (GD) cannot be relied on, as it gets stuck in the local maxima closest to A_0. RF does not blindly climb the closest steepest hill, but instead its sight extends, on 6 rays, until the boundaries of Ω. Therefore, at every iteration, it gets another chance to notice a higher hill elsewhere and will eventually jump on it. The new hill is climbed until its tip, or until a searching ray intersects an higher hill, and so on. The chances of finding the absolute maximum are much higher than for GD and are fairly high on absolute in the kind of spaces we consider, which are highly non-convex, but also not very large (R_2 roughly corresponds to R_1). Figure 2 exemplifies RF's behaviour in 2 dimensions and shows a typical case where it succeeds while GD fails.

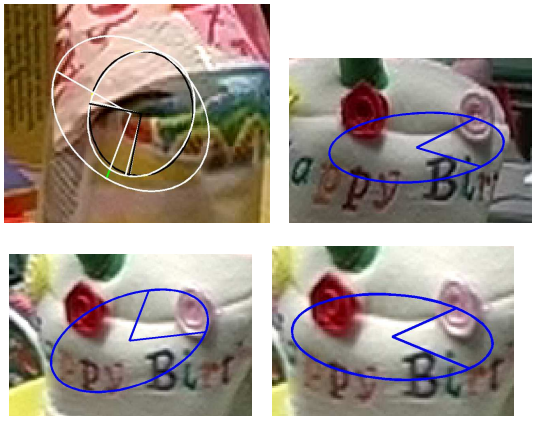

Figure 3: *Top-left: The small white region has been artificially deformed by simultaneous translation, rotation, scale and shear (*$A = (-5, 0, 1.4, 1.3, Pi/7, 0.4)$*) to the large white one. The refined region (black) comes very close to the correct solution. Top-right: region in a view. Bottom-left: match before refinement, right: refined.*

In practical situations RF iterates 3 to 10 times. Given the search space boundaries and discretisation steps above, this amounts to 100 to 700 evaluations of the similarity measure. Thus, RF is about as fast as GD and is orders of magnitude faster than exhaustive search (3.5 mio for the same parameters) or simulated annealing. This makes RF computationally affordable for our purposes, where thousands of refinements are required.

The refinement algorithm is applied to all region-tracks $R \in \Gamma$ provided by the conflict-resolution algorithm (section 2) so as to obtain a *globally* optimal registration. To achieve this, all regions of R are refined towards the same pivot-view. The *pivot-view* is selected as the one maximising the sum of similarities to all others views where R is present. The sum of similarities for each view is computed after refinement to all other views. Note it is highly unlikely that a view where R is mismatched will be selected as pivot. This simple approach is independent of the order in which the views are considered and makes all regions within a region-track well globally aligned. A high quality registration is important for the following propagation stage.

4. Propagation

As pointed out in section 1, many regions of a view V_1 do not get matched to another view V_2 even though the feature is visible in the image (e.g.: the corresponding region has not been extracted, or maybe it has been extracted but the matching failed). This section describes an approach for exploiting the information supplied by a correct match in order to generate many other correct matches. Consider a region C_1 in V_1 without correspondent in V_2 (*candidate*

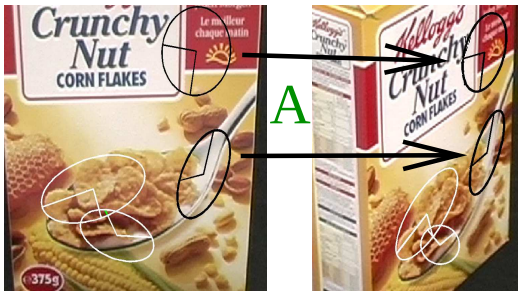

Figure 4: *The candidate region (spoon) is propagated to the right view via the affine transformation (A) of a support region (top).*

region) and a set $\Psi = \{S_1^i\}$ of matched *support* regions in the neighborhood [2] of C_1 (figure 4). If S_1^i and C_1 lie on the same physical surface (e.g.: a facet of an object), then they will probably be mapped to V_2 by similar affine and photometric transformations. For every $S_1^i \in \Psi$ do:

1. Compute the affine transformation A^i mapping S_1^i to S_2^i in view V_2.
2. Compute the color transformation $T_{RGB}^i = \{s_R, s_G, s_B\}$ between S_1^i and S_2^i. This is composed by the scale factors on the three colorbands.
3. Project C_1 to view V_2 via $A^i : C_2^i = A^i C_1$.
4. Evaluate the similarity between C_2^i and C_1 after applying the color transformation T_{RGB}^i

$$sim_i = \mathrm{NCC}(T_{RGB}^i C_1, C_2^i) + (1 - \frac{\mathrm{dRGB}(T_{RGB}^i C_1, C_2^i)}{100})$$

Applying T_{RGB}^i allows us to use the unnormalized color-distance dRGB on the raw image pattern, because color changes (e.g.: darker light) are now compensated for. This is an advantage as it provides maximal discriminative power.

We retain C_2^{best}, with $best = \arg\max_i sim_i$, the region that best matches C_1 and refine it by the algorithm of section 3, yielding C_2^{ref}. This refinement step adapts C_2^{best} to the local plane orientation and counters perspective effects (the affine approximation is valid only on a local scale). C_2^{ref} is considered correctly propagated if $\mathrm{sim}(T_{RGB}^i C_1, C_2^{ref}) > t_{prop}$ ($t_{prop} = 1.0$ in our experiments). Note that refinement tends to raise the similarity of correctly propagated regions much more than the similarity of mispropagated ones, bringing an important benefit: the increase in the *separation* between the distributions of the similarity for mispropagated regions and for correctly propagated ones. Extensive experiments have shown the last thresholding step to be remarkably more effective after refinement.

[2] In all experiments a circle of radius $\frac{1}{5}$ of the image width has been used

Note that this approach *generates* a new region in V_2 which might not have been originally extracted (differently than in [5], where the geometric transformation is used only to guide the search for further matches). This is important as it helps solving the main problem exposed in section 1: the quick drop of the probability that a region is extracted simultaneously in several views. Indeed propagation strongly increases the chances that a region will be put in correspondence, as it suffices that any nearby region undergoing a similar image transformation is correctly matched.

For every pair of views $l, m, l \neq m$, the propagation algorithm is applied to all candidate regions $\Phi_l \backslash \Phi_m$ which are present in l, but not in m, using as support the regions Φ_{lm} already matched between the two views. Everytime a region R_l^i is successfully propagated to view m, it is added to region-track R^i. Propagation doesn't generate new region-tracks, but *extends* currently existing ones. As consequence, the region-tracks grow larger (i.e.: they are present in more views) and the connectedness between any subset of views vs, i.e.: the amount $|\Phi_{vs}|$ of region-tracks present simultaneously in each $v \in vs$, is strongly increased (as shown in section 6). Note also that a single propagation creates many new pairwise matches, as all regions in the track are implicitly assumed matched to the newly added region. These *transitive propagations* contribute actively to increasing the inter-view connectedness.

5. Topological filter

The region matches along the views might still contain some mismatches. These are due to original mismatches that survived the conflict-resolution stage and occasional mispropagations constructed on them. A new method for removing mismatches based on a topological constraint for triples of regions is introduced.

5.1 The sidedness constraint

Consider a triple (R_1^1, R_1^2, R_1^3) of regions in V_1 and their corresponding regions (R_2^1, R_2^2, R_2^3) in V_2. Let $\mathbf{c_v^i}$ be the center of region R_v^i. The function

$$\mathrm{side}(R_v^1, R_v^2, R_v^3) = \mathrm{sign}((\mathbf{c_v^2} \times \mathbf{c_v^3})\mathbf{c_v^1}) \quad (3)$$

takes value -1 if c_v^1 is on the right side of the directed line $\mathbf{c_v^2} \times \mathbf{c_v^3}$ from c_v^2 to c_v^3, or value 1 if it's on the left.
The equation

$$\mathrm{side}(R_1^1, R_1^2, R_1^3) = \mathrm{side}(R_2^1, R_2^2, R_2^3) \quad (4)$$

states that $\mathbf{c^1}$ should be on the same side of the line in both views (figure 5). The *sidedness constraint* (4) holds for all triples of coplanar regions, because in this case property (3) is viewpoint invariant. Equation (4) happens to be valid also for most non-coplanar triples. A triple for which equation

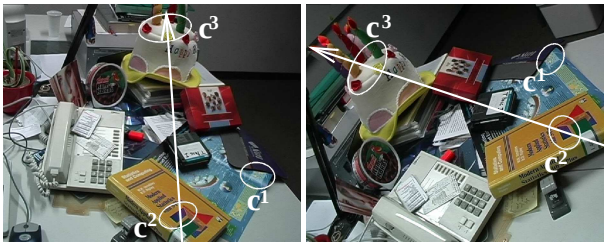

Figure 5: *Sidedness constraint.* $\mathbf{c^1}$ *should be on the same side of the directed line going from* $\mathbf{c^2}$ *to* $\mathbf{c^3}$ *in both views.*

(4) doesn't hold is said to *violate* the constraint. This happens when at least one of the three regions is mismatched, or when the regions are not coplanar and there is important camera translation in the direction perpendicular to the 3D scene plane containing their centers. The latter can create a parallax effect strong enough to move $\mathbf{c^1}$ to the other side (*parallax-violation*). However, this happens only to a small minority of triples, for any given pair of views, as also confirmed by our experiments. Region-tracks R^i, R^j, R^k violate or respect equation (4) independently of the order in which they appear in the triple. The three points should be cyclically ordered in the same orientation (clockwise or anti-clockwise) in the two images in order to satisfy (4).

5.2 Removing mismatches

A triple including one, or more, mismatched regions has higher chances to violate the (4). When this happens we can only conclude that at least one of the regions in the triple is mismatched, but we do not know yet which. While one triple is not enough to decide, this information can be recovered by considering all triples simulteneously. By integrating the weak information each triple provides, it's possible to robustly discover mismatches. The key idea is that incorrectly located regions will be involved in a higher share of violations. [8] already noted the benefits of analysing topological configurations of points and lines.

Equation (4) is checked for all triples of regions $(R^i, R^j, R^k), R^i, R^j, R^k \in \Phi_{12}$, with Φ_{12} the set of all regions present in both V_1, V_2. Let $\Phi = \{i | R^i \in \Phi_{12}\}$. The algorithm starts by computing

$$\mathrm{h}(i) = \sum_{j,k \in \Phi \setminus i, j>k} |\mathrm{side}(R_1^i, R_1^j, R_1^k) - \mathrm{side}(R_2^i, R_2^j, R_2^k)| \quad (5)$$

the amount of violations R^i is involved into, for all $i \in \Phi$. $\mathrm{h}(i)$ is then normalized w.r.t. the total amount of possible violations any single region can be involved into

$$\mathrm{h_N}(i) = \frac{\mathrm{h}(i)}{(n-1) * (n-2)}, \quad n = |\Phi|.$$

The most violating region R^w, with $w = \arg\max_i \mathrm{h_N}(i)$ is determined. If $\mathrm{h_N}(w) > t_{topo}$, region R^w is considered a mismatch and removed from Φ. At each iteration $\mathrm{h_N}(i)$ is recomputed based on the remaining regions in Φ and eventually the most violating region is removed. The process iterates until no more regions can be removed. It's wise to store the terms of the sum $\mathrm{h}(i)$ during the first iteration. In the other iterations, $\mathrm{h}(i)$ can be quickly recomputed by retrieving and adding up the necessary terms, making the computational cost almost independent on the amount of iterations.

During the first iterations, when several mismatches are still within Φ, even correctly matched regions might have a high $\mathrm{h_N}$, because of their participation in triples including a mismatch. However, the mismatched regions will have an even higher $\mathrm{h_N}$, because they will be involved in the very same triples, plus other violating ones. Thus the worst mismatch R^w, i.e.: the region which is located in V_2 farthest from where it should be, has the highest chance to violate each individual constraint and therefore will have the highest $\mathrm{h_N}$. Once R_w is removed, all $\mathrm{h_N}$ will decrease, and the second worst mismatch will have the highest value. When only correct matches are left, small error percentages due to occasional parallax-violations are still reflected by $\mathrm{h_N}$. However, these will probably be lower than t_{topo}, causing the algorithm to stop.

Various experiments on artifical and real configurations confirmed these theoretical considerations. The algorithm proved very robust, by withstanding large amounts of mismatches. In experiments with various scenes, up to 65% of the regions where translated to uniformly distributed random locations. The algorithm successfully removed all relocated regions, while losing at most a few correct matches.

If all region-tracks of a triple are uniformly random distributed on the two images, then $\mathrm{h_N}(i)$ has expected value 0.5 for the three. This, together with experimental measurements on mismatch-free configurations, helped us selecting $t_{topo} = 0.15$.

The topological filter is applied to all pairs of views $l, m, l < m$ yielding a set $\{(R_l, R_m)\}$ of mismatches per pair. A single mismatch still doesn't tell which one of R_l or R_m is wrong. However, this information can be infered from the set of mismatches related to a single region-track (i.e.: if R_l is involved in 5 mismatches and R_m in only 1). Removing either R_l or R_m suffices to eliminate the mismatch. We remove the minimum amount of regions from each region-track, such that all mismatches are eliminated. This approach eliminates all detected mismatches, while minimally reducing the region-tracks, thus enhancing quality without sacrificing inter-views connectedness.

6. Results and conclusion

The multi-view matching scheme has been tested on many sets of images (scenes). All reported scenes have been pro-

Figure 6: *Valbonne. 230 3-view matches on views* $9, 12, 14$.

cessed with the same set of parameters reported in the paper.

In the first example 10 images (ids 1,2,3,5,9,11,12,13,14,15) from the *Valbonne* scene (6), were processed. This image set poses several challenges, like significant viewpoint differences and uniform colors and textures. Table 1 shows the number of region-tracks present in various subsets of views. Note the high entries for 4 views and more, indicating strong connectedness between views (e.g.: about 200 4-view matches). The statistical problem stated in section 1, namely the rapid decrease of the chance of obtaining a N-view track, with increasing N, has been remedied: the amount of N-view matches gracefully decreases (close to linearly) with increasing N and corresponds well with failing visibility. The third column (CR) reports the amount of matches just after forming the tracks, via the algorithm proposed in section 2. At this stage, the tracks are still vulnerable, and their number decays exponentially (e.g.: 51 2-view matches in $(9, 11)$ roughly half in $(9, 11, 12)$ and only 3 in $(3, 5, 9, 11, 12, 13)$). However, the first stage's main goal is to solve the structural problem, by formatting the data in region-tracks. It's the combined effect of refinement, propagation and topological filtering that then counters the statistical problem, as demonstrated in the second column of the table. The increase in the amount of N-view matches is strong, and many of them now exist even when *none* could be found in the original data (from 9 views on). The values in table 1 compare favorably against the ones reported in [5] , corroborating the effectiveness of our approach. The correctness of the tracks Φ_{vs} present simultaneously in a set of views vs is evaluated as $1 - \frac{errors}{|\Phi_{vs}|*(|vs|-1)}$, where $errors$ is the total amount of incorrectly located regions, computed over all tracks and views. This measure ranges from 0 (all mismatches) to 1 (perfect tracks), and takes into account that a track can be *partially* correct, if only some of its regions are mis-located. Figure 6 shows 230 tracks (96% correct), distributed over the whole church, in views $(9, 12, 14)$.

The *Birthday* scene features a more complex geometry than Valbonne and more diverse textures. Eight very different viewpoints serve as input (ids 1 to 8). Figure 7 shows 135 tracks (95% correct) on 3 views. Two of the views are

views	tracks	CR	views	tracks	CR
9 11	340	51	all 10 views	11	0
9 11 12	295	26	9 12 14	230	13
5 9 11 12	204	9	3 9	273	39
3 5 9 11 12 13	153	3	3 5 9	254	21
3 5 9 11 12 13 14	100	2	3 5 9 11	208	7
2 3 5 9 11 12 13 14	69	1	3 5 9 11 15	132	5
1 2 3 5 9 11 12 13 14	29	0	2 3 5 9 11 15	124	3

Table 1. Number of tracks for Valbonne.

Figure 7: *Birthday. Left: top to bottom: views* $1, 8, 7$. *Right: A 5-view track.*

taken from almost opposite directions $(7, 8)$. Nevertheless, the matches well cover the commonly visible parts of the scene. The telephone, and the picture of the girl above it, are affected by strong out-of-plane rotation, but still have several correct matches. 124 region-tracks (98% correct) are present in another four images (figure 8). Note the quality of the tracks on the mousepad, which undergo strong image scaling and rotation (views $2, 6$) , and the lack of 4-view matches on the book. This is correct, as it's not visible in view 4. Robustness to scale changes is demonstrated in figure 9, where viewpoint 5 is significantly closer to the scene than 4. Nevertheless, the two cameras see largely the same part of the scene, and the system produces 178 tracks (97% correct), densely covering the images.

Table 2 summarizes the large increase in N-view matches

views	tracks	CR	views	tracks	CR
1 6	241	48	all 8 views	46	0
1 4 6	185	14	2 8	170	25
1 2 4 6	124	4	2 3 8	161	16
1 2 3 4 6	108	2	2 3 7 8	120	3
1 2 3 4 6 8	81	1	2 3 4 7 8	93	1
1 2 3 4 6 7 8	79	0	1 2 3 4 7 8	81	0

Table 2. Number of tracks for Birthday.

Figure 8: *Birthday. Top: views* $1, 2$*. Bottom: views* $4, 6$*.*

brought by the method. The amount of N-view matches decreases close to linearly with N. This is particularly significant in this scene, where the parts visible in all N-views also decrease significantly.

The accuracy of the registration within a region-track is exemplified in figure 7. The ellipse's shape and orientation cover the same physical surface in all views, accurately deforming to fit the viewpoints. This entails rotation, skew and anisotropic scale changes, largely covering the affine spectrum. Figure 9 depicts another interesting case, where a region gradually rotates over almost 180 degrees out-of-plane during 5 views. These come from a scene featuring a 19 views tour around a cereal box. The good alignment is made possible, and computationally affordable, by the new optimization technique presented in section 3.

The experiments confirm that the presented approach can generate a large amount of high quality region tracks starting from an unordered set of images taken from substantially different viewpoints. The tracks strongly connect the views and provide a good coverage of the commonly visible parts of the scene. This is particularly important in object-recognition, where more coverage means a more complete description of the object. Accuracy is more an issue in 3D-reconstruction and has also been improved. The method works in conjunction with most WBS algorithms. Future developments include the application to object-recognition and efficiency improvements.

Figure 9: *Top: view 4. Middle: view 5. Frame: 5 views from a cereal-box track.*

References

[1] T. Tuytelaars and L. Van Gool Wide Baseline Stereo based on Local, Affinely invariant Regions *British Machine Vision Conference*, pp. 412-422, 2000.

[2] K.Mikolajczyk and C.Schmid An affine invariant interest point detector *ECCV*, vol. 1, 128–142, 2002.

[3] A.Baumberg Reliable Feature Matching Across Widely Separated Views *Intl. Conf. on Comp. Vis.*, pp. 774-781, 2000.

[4] J. Matas, O. Chum, M. Urban and T. Pajdla Robust Wide Baseline Stereo from Maximally Stable Extremal Regions *British Machine Vision Conf.*, pp. 414-431, 2002.

[5] F. Schaffalisky and A. Zisserman Multi-view matching for unordered image sets *European Conf. on Comp. Vis.*, 2002.

[6] D. Lowe Object Recognition from Local Scale-Invariant Features *Intl Conf. on Comp. Vis.*, pp. 1150-1157, 1999.

[7] D. Tell and S. Carlsson Wide baseline point matching using affine invariants *ECCV*, vol. 1, pp. 814-828, 2000.

[8] S. Carlsson Combinatorial Geometry for Shape Representation and Indexing *Obj. Repr. in Comp. Vis. II*, Ponce, Zisserman eds. 1996