Chapter 1

Multi-View Surface Reconstruction by Quasi-Dense Wide Baseline Matching

Juho Kannala, Markus Ylimäki, Pekka Koskenkorva, and Sami S. Brandt
Machine Vision Group
Computer Science and Engineering Laboratory
University of Oulu, Finland
{*jkannala,ylimakma,pkoskenk,sbrandt*}*@ee.oulu.fi*

This chapter provides a review on correspondence growing techniques which have been used in multi-view stereo reconstruction problems. Typically these methods approach the problem of multi-view image matching by first determining a sparse set of feature correspondences between pairs of views and then iteratively expanding the matching regions. Sometimes such techniques are also referred by terms like match propagation, quasi-dense matching, or surface growing. Besides providing an overview of the research area, this chapter introduces a particular method, called quasi-dense wide baseline matching, which employs the best-first correspondence growing principle for matching pixels in views with substantially different viewpoints. In addition, the properties and performance of different methods are illustrated by examples and experiments with real images.

1. Introduction

Automatically acquiring a three-dimensional model of a scene from multiple photographic images is an important research area in computer vision. The basic geometric and computational principles for automatic multi-view reconstruction systems have been known for some time [20], and several such systems have already been built. The first systems of this kind were designed to use continuous video sequences as their input whereas some of the more recent approaches are able to acquire scene reconstructions from wide baseline image sets, where the views are captured at sparsely located viewpoints [36], or even unorganized image sets downloaded from Internet photo collections [48; 17].

However, despite the large number of previous research efforts and existing reconstruction systems, there are still many challenges and open problems in the field. In fact, image-based modeling continues to be an active research area. For example, one current focus area is the construction of large-scale reconstruction systems, which are able to reconstruct city-scale scenes from thousands or hundreds of thousands of images [1; 13; 12]. In addition, there have been efforts to improve

the efficiency of video-based reconstruction pipelines in order to achieve real-time performance [42; 40; 39]. Also, one important research topic is to develop methods for representing three-dimensional models in a compact form that facilitates storage and transmission [32; 6].

The great interest towards image-based modeling is partly motivated by the recent trends in information and communication technology industry. For instance, in many applications and Internet-based services, there is an increased need for three-dimensional photorealistic visualization of scenes and objects, possibly with additional spatially localized data. Examples of such applications include navigation and driving assistance, entertainment and virtual tourism, architectural and environmental planning, and various forms of personal communication. In fact, image-based models are already utilized in many recent services by major software companies, such as Photosynth image stitching tool by Microsoft, and geographical mapping and visualization tools like Google Earth and Bing Maps.

In this article, we concentrate on image matching which is an essential part of any generic multi-view reconstruction system. Specifically, we focus on a particular correspondence growing method called *quasi-dense wide baseline matching* that was originally proposed in the articles [24] and [29] on which this chapter is partly based. This method tries to establish a large number of point correspondences, i.e. a quasi-dense set of matching points, between two or three views of a scene. As its input the algorithm takes a sparse set of corresponding regions between pairs of views and then iteratively expands these regions pixel-wise by using a best-first match propagation strategy similar to [33].

The quasi-dense approach can be used to match pairs of views of arbitrary possibly deforming scenes if the surfaces of the scene are sufficiently textured [25]. However, if the fundamental matrix is known for a pair of perspective views of a rigid scene, the method [24] can also utilize that to improve the reliability of matching. Further, in the three-view case, the method of [29] can be used if the trifocal tensor is known. In this chapter we mainly concentrate on cases where the cameras are perspective and their projection matrices are known. In fact, this is the usual problem setting in multi-view stereo [45], where the main task is to acquire a dense or semi-dense reconstruction by matching pixels between multiple calibrated views. Multi-view stereo is a key stage in a typical 3-D reconstruction pipeline after the camera motion and sparse scene structure have been estimated by structure from motion techniques [20; 16].

The structure of this chapter is as follows. First, in Section 2 we review related work on correspondence growing methods and multi-view reconstruction. Then, in Section 3, we describe the quasi-dense matching algorithms for image pairs and triplets and also discuss possible generalizations in order to match multi-view image sets with more than three views. Section 4 presents illustrative examples and experimental results with real images. The results are discussed in Section 5 and Section 6 concludes the chapter.

2. Related work

Matching multiple views of a scene in order to obtain a reconstruction of it is an old [20] but still timely problem in computer vision. Perhaps the most well studied topic in this area is two-view stereo matching. Much of the early research on two-view stereo was concentrated on the narrow baseline case [43] but some more recent works [24; 52] have focused on matching widely separated views by building upon the recent techniques in sparse wide baseline matching [38]. However, although some image-based modeling approaches use two-view matching as a basic building block and then combine point clouds from pairwise stereo depth maps into complete object models as a post process, there are also many multi-view stereo methods which are inherently designed to process more than two views [45; 14]. In the following, we aim to give an overview of two-view and multi-view stereo reconstruction methods with a particular focus on correspondence growing methods that are related to our approach.

2.1. *General overview*

Multi-view stereo methods can be characterized according to the scene representation that they use. Most approaches to represent three-dimensional scenes are based on voxels [30; 44], level-sets [11], polygonal surface meshes [10; 21], depth maps [49; 16; 15] or point clouds [14; 34]. Typically level set methods and voxel-based approaches represent scene geometry as a function on a regularly sampled 3-D grid. That is, in the case of level sets, the function encodes distance to the closest surface and, in voxel representation, it is a simple discrete occupancy function. The main problem with regular 3-D grids is their high memory usage which makes them inefficient for high-resolution representation of large scenes.

In contrast, polygonal meshes are efficient to store and render since they represent surfaces as a set of connected planar facets that may have unconstrained position in space. For example, triangular meshes have been used to model large scenes in some recent works [21]. Nevertheless, sometimes a depth map for each input view is the most convenient and natural representation for 3-D information. For example, in a real-time reconstruction system that operates from a moving vehicle, an efficient plane-sweeping method can be used to produce depth maps sequentially from successive views [15; 42]. However, multi-view depth maps are not necessarily mutually consistent and, in the end, some kind of a fusion process is often used to combine multiple depth maps into a single point cloud or surface mesh [37; 31]. In fact, besides meshes, point clouds (or clouds of surface patches) are representations that are also commonly used [14; 27]. Still, in many cases point clouds are eventually transformed into meshes in order to facilitate efficient storage and rendering [32; 6]. Finally, it is also common that systems use different 3-D representations in different stages of the reconstruction pipeline.

Besides scene representation, another essential characteristic of multi-view stereo

methods is the type of reconstruction algorithm. In fact, reconstruction algorithms can be categorized into global and local methods. Global methods typically define a global cost function for shapes and then use some global optimization algorithm to recover a shape that minimizes the cost function. Examples of global methods include approaches based on volumetric Markov Random Field (MRF) models [3], which utilize graph cut optimization techniques [2], and variational approaches, which use convex optimization [28]. A common property of global methods is a relatively large memory usage and time complexity due to the use of a dense volumetric grid. Hence, such methods are not particularly suitable for large-scale scenes. However, there are also works which improve efficiency of graph cut based approaches by using adaptive grids [47; 31].

On the other hand, local matching methods often allow faster reconstruction with smaller memory requirements by, for example, dividing large image sets into subsets which may be processed separately and partly in parallel [42]. However, this may increase reconstruction errors in scene regions that are not particularly distinctive. That is, the global fitness of reconstruction may be compromised. Thus, the erroneously reconstructed parts need to be improved by some post-processing approach, using e.g. depth map fusion [42] or other ways of enforcing visibility consistency [14].

There are various local matching methods, including approaches based on the plane-sweep algorithm [8; 15] or correspondence growing [33; 14], for instance. Often these local approaches use scene representation based on surface patches, point clouds, or depth maps, which are flexible, as they can easily model various topologically complex structures and do not need a bounding box or a visual hull of the scene for initialization.

In addition, besides approaches that can be clearly categorized as local or global, there are semi-global methods [22], which do not directly aim in finding a global optimum of a global cost function but still consider certain long-range interactions between matched scene points (instead of just using their immediate neighborhood as in purely local methods). For example, the commonly used dynamic programming approach for narrow-baseline two-view stereo matching [43] can be seen as a semi-global method.

Overall, it should be noted that a complete multi-view reconstruction system may use different kinds of algorithms in different stages. For example, the recent approach [21] uses local matching of input views to extract a dense but possibly redundant point cloud, then applies a global approach for meshing the point cloud, and finally refines the resulting mesh model by iterative local optimization of a global cost function.

2.2. *Correspondence growing methods*

In the following sections we concentrate on a local image matching approach [24; 29], which we call quasi-dense wide baseline matching and which iteratively expands corresponding image regions by the match propagation algorithm [33]. However, somewhat similar ideas of correspondence growing have been used in other works as well, as briefly reviewed below.

One of the first correspondence growing methods is [41], which gradually expands matching image patches in a pair of views by using iterative alignment of patches [18] during each expansion step. Like [41], our approach uses the best-match-first growing stategy but we do not perform iterative refinement of the matched patches during growing. Also, [41] applies only for pairs of views whereas [29] extends the best-first matching approach to triplets of views if the trifocal tensor is known.

Our growing algorithm is similar to the match propagation algorithm of [33], which also avoids iterative patch refinements during growing and imposes a uniqueness constraint (i.e. one-to-one matching) and a disparity gradient limit simultaneously. However, unlike [33], our implementation [24] can be directly used for wide baseline image pairs. This is achieved by using an affine transformation model for the local patches instead of a translational model.

Another relatively recent correspondence growing method is [5], which is also inspired by [33]. However, unlike [33] and [24], [5] solves a global optimization task, which allows to reduce matching errors and prevents matching ambiguous structures, e.g. due to repetitive texture patterns. Nevertheless, the implementation in [5] requires a rectified stereo image pair as input and, hence, it can not be used for images of nonrigid scenes. Also, in practice, the ambiguities caused by repetitive patterns can be reduced by using more than two views, and even greedy local matching [24] may perform well in such cases [29].

Recently, correspondence growing algorithms have also been used for true multi-image matching with datasets that contain more than two or three views [19; 14]. These methods expand surface patches in 3-D space so that the patches are directly matched between multiple views. The advantage of such patch-based approaches is their flexibility in modeling both small, compact objects and large, complex scenes, and even crowded scenes where moving obstacles appear in multiple images of a static structure of interest [14].

Perhaps one of the most widely used patch-based multi-view stereo methods is [14], which is publicly available in source form and uses repeated match expansion and filtering stages for reconstruction. The method has produced good results with benchmark datasets [45] and it has also been used for large datasets [13]. However, [14] does not use a best-first growing strategy as we do. In fact, our results indicate that the best-first strategy allows to acquire good matches with a single growth stage and without repeated expansion and filtering steps. Hence, one key aspect in this chapter is to discuss the possibilities to further advance the use of correspondence growing methods for direct multi-view matching that is both accurate and efficient.

3. Quasi-dense wide baseline matching

In this section, we describe the two-view and three-view quasi-dense matching algorithms which were originally proposed in [24] and [29], respectively. The algorithms are based on the match propagation algorithm [33] which is extended to be applicable for wide baseline images whose viewpoints differ substantially.

3.1. *Two-view matching*

Given two views, $\mathcal{I}_1$ and $\mathcal{I}_2$, and optionally the associated fundamental matrix $\mathcal{F}_{21}$, our matching approach produces a quasi-dense set of point correspondences between the two views by growing a sparse set of seed matches, which are determined by matching affine covariant regions [24; 38].

Hence, the method contains two stages: the initial matching stage and the growth stage. The output of the initial matching stage is a set of seed matches $\{\mathbf{s}^i\}_i$, where each seed contains image coordinates $\mathbf{x}_a$ and $\mathbf{x}_b$, which denote the centroids of the matched regions [38], and an affine transformation matrix $\mathbf{A}_{ab}$, which approximates the local geometric transformation between the views. For each seed, the index $a \in \{1, 2\}$ indicates the reference view, and b is the other view. The reference view is determined so that the affine transformation $\mathbf{A}_{ab}$ from a to b is magnifying, i.e. $|\det \mathbf{A}_{ab}| \geq 1$ [24]. Further, each seed is associated with a texture similarity score s_{ab} and intensity variance score v. The zero-mean normalized cross-correlation (ZNCC) of geometrically normalized image patches is used as the similarity measure, and the score v is set equal to the minimum intensity variance of the two patches. The data structure for the two-view seeds is summarized in Definition 1. The same structure is used also for the grown matches.

In the growth stage, the seeds are sorted into a priority queue $\mathcal{Q}$ according to their similarity scores and then propagated by iterating the following steps:

(i) The seed $\mathbf{s}$ with the best score is removed from $\mathcal{Q}$.
(ii) New candidate matches are searched nearby $\mathbf{s}$ by using $\mathbf{s}.\mathbf{A}_{ab}$ for the geometric normalization of local image patches.
(iii) The candidates, which have a sufficiently high similarity and which satisfy the disparity gradient limit and the epipolar constraint (optional), are added to $\mathcal{Q}$ and to the list of matches after updating their affine transformation estimates. The corresponding pixels in the matching tables are marked as reserved.

The geometric normalization of patches and the update of affine transformations are detailed in [24]. Further, it should be noted that, for each grown match, the indices a and b are determined from the updated transformation matrix, i.e., the role of views 1 and 2 may be swapped during propagation. The outline of the growth stage is described in pseudo-code in Algorithm 1.

Updating affine transformation parameters for new matches allows the propagation to adapt to variations in the orientation and pose of surfaces as the matching

Definition 1: Data structure for two-view seed matches

```
struct twoviewseed{ int a, b;
                    double x_a, x_b, A_ab, s_ab, v; };
```

Algorithm 1: Two-view match propagation

Input: images $\mathcal{I}_1$, $\mathcal{I}_2$, two-view seed matches $\mathcal{S}_{12}$, thresholds ϵ_d, ϵ_e, t, t_u, z, z_u, and, optionally, fundamental matrix $\mathcal{F}_{21}$

Output: list of matches $\mathcal{M}$, matching tables $\mathcal{J}_1$, $\mathcal{J}_2$

Initialize $n=0$, $\mathcal{M}=\emptyset$, $\mathcal{J}_k(\mathbf{p})=0$ for all k, $\mathbf{p}$
Compute pairwise similarity scores $\mathbf{s}.s_{ab}$ and uniformity scores $\mathbf{s}.v$ for all seeds $\mathbf{s}$, $[\mathbf{s}.s_{ab}, \mathbf{s}.v] = \mathtt{sim}(\mathbf{s}, \mathcal{I}_{\mathbf{s}.a}, \mathcal{I}_{\mathbf{s}.b})$
Sort the seeds according to the scores $\mathbf{s}.s_{ab}$
Initialize priority queue $\mathcal{Q}$ with sorted seeds
while $\mathcal{Q}$ not empty
    Draw the seed $\hat{\mathbf{q}} \in \mathcal{Q}$ with the best score $\hat{\mathbf{q}}.s_{ab}$
    Set $a=\hat{\mathbf{q}}.a$ and $b=\hat{\mathbf{q}}.b$
    (In the following, $\mathcal{I}_a$ and $\mathcal{I}_b$ define new seeds)
    for each new match $\mathbf{q}^i$ nearby $\hat{\mathbf{q}}$ which satisfies the disparity gradient limit ϵ_d and, optionally, the epipolar constraint ϵ_e
        Set $\mathbf{q}^i.s_{ab} = -\infty$
        if $\mathcal{J}_a(\mathtt{round}(\mathbf{q}^i.\mathbf{x}_a))=0$ & $\mathcal{J}_b(\mathtt{round}(\mathbf{q}^i.\mathbf{x}_b))=0$
            $[\mathbf{q}^i.s_{ab}, \mathbf{q}^i.v] = \mathtt{sim}(\mathbf{q}^i, \mathcal{I}_a, \mathcal{I}_b)$
    end for
    Sort matches $\mathbf{q}^i$ according to the scores $\mathbf{q}^i.s_{ab}$
    for each $\mathbf{q}^i$ satisfying $\mathbf{q}^i.s_{ab} \geq z$ and $\mathbf{q}^i.v \geq t$
        Set $n=n+1$
        if $\mathbf{q}^i.s_{ab} \geq z_\mathrm{u}$ and $\mathbf{q}^i.v \geq t_\mathrm{u}$
            Update $\mathbf{q}^i.\mathbf{A}_{ab}$, and thereafter $\mathbf{q}^i.a$ and $\mathbf{q}^i.b$
        Set $\mathcal{Q}=\mathcal{Q}\cup\{\mathbf{q}^i\}$ and $\mathcal{M}=\mathcal{M}\cup\{\mathbf{q}^i\}$
        Set $\mathcal{J}_a(\mathtt{round}(\mathbf{q}^i.\mathbf{x}_a))=n$, $\mathcal{J}_b(\mathtt{round}(\mathbf{q}^i.\mathbf{x}_b))=n$
    end for
end while

expands further from seed regions. The update of the affine transformation matrix is implemented via a simple non-iterative update rule which is based on local second-order intensity moment matrices of the images [24]. In addition to intensity moments, the update rule requires a pair of corresponding directions in the images. Such directions can be obtained from the fundamental matrix or, if it is not available, they can be estimated from local image gradients [25]. In the latter case, the

match propagation may even be used to track non-rigid deformations of surfaces [25]. The frequency of affine adaptation is controlled by parameters z_{u} and t_{u} in Algorithm 1 which restrict the update to occur only for textured image patches whose similarity scores exceed a threshold z_{u}. If $z_{\mathrm{u}} > 1$, the update is omitted always and the algorithm operates in a non-adaptive mode.

In summary, as shown in Algorithm 1, the result of two-view match propagation is a list of grown matches $\mathcal{M}$ and two matching tables $\mathcal{J}_1$ and $\mathcal{J}_2$, which have the same size as images $\mathcal{I}_1$ and $\mathcal{I}_2$, respectively. The nonzero values in $\mathcal{J}_1$ and $\mathcal{J}_2$ indicate the pixels which are nearby to the sub-pixel matches of $\mathcal{M}$. That is, the closest pixel to a given coordinate vector $\mathbf{x}$ is $\mathbf{p} = \texttt{round}(\mathbf{x})$, and a nonzero value $\mathcal{J}_k(\mathbf{p})$ is an index to the corresponding item in $\mathcal{M}$. The value $\mathcal{J}_k(\mathbf{p}) = 0$ indicates that pixel $\mathbf{p}$ is not matched.

3.2. *Three-view matching*

The three-view matching approach [29] builds on the ideas of the two-view method [24] but there are some additions and modifications which improve the robustness of matching for view triplets when the trifocal tensor is known. Thus, although the two-view method can be used without knowing the fundamental matrix, the three-view method always requires the trifocal tensor as input. Another difference to the two-view method is that the ordering of seeds in the priority queue $\mathcal{Q}$ is based on a total score s which combines two pairwise similarity scores, s_{ab} and s_{ac}, between the reference view a and the other two views. Hence, at each propagation step, the seed with the best total score is grown.

The three-view method is shown in pseudo-code in Algorithm 2, which is quite similar to Algorithm 1. However, there are certain additional steps (i.e. lines 3, 4, 5, 21, 22, 25, and 26), which are detailed in the following. The data structure for three-view matches is given in Definition 2.

First, since the input to Algorithm 2 is a set of two-view seeds, as in the two-view method, the seeds have to be transformed to three-view seeds by using trifocal transfer [20]. That is, we define a function called `transfer`, which transforms a match $(\mathbf{x}_a, \mathbf{x}_b)$ to the third view, indexed by c, and also computes the local affine transformation $\mathbf{A}_{ac}$ between the reference view a and the view c (line 3 in Algorithm 2). The function is implemented so that it uses $\mathbf{x}_a$, $\mathbf{x}_b$ and $\mathbf{A}_{ab}$ to define three corresponding points in the views a and b and then transforms this local affine basis to the third view by trifocal transfer [20]. Thereafter, $\mathbf{A}_{ac}$ may be solved from the three point correspondences between a and c. Finally, given $\mathbf{x}_c$ and $\mathbf{A}_{ac}$, one may also evaluate the local similarity s_{ac} between views a and c.

Given a three-view seed $\mathbf{s}$, the total score $\mathbf{s}.s$, on which the ordering in $\mathcal{Q}$ is based, combines the pairwise similarities between a and the other two views. This allows the three-view method to perform better than a combination of pairwise

Definition 2: Data structure for three-view seed matches

`struct threeviewseed{` `int` a, b, c;
`double` $\mathbf{x}_a$, $\mathbf{x}_b$, $\mathbf{x}_c$, $\mathbf{A}_{ab}$, $\mathbf{A}_{ac}$, s_{ab}, s_{ac}, s, v; `}`;

Algorithm 2: Three-view match propagation

`Input:` images $\mathcal{I}_1$, $\mathcal{I}_2$, $\mathcal{I}_3$, two-view seeds $\mathcal{S}_{12}$, $\mathcal{S}_{13}$, $\mathcal{S}_{23}$, thresholds ϵ_d, ϵ_e, t, t_u, z, z_u, $\tilde{z}$, and trifocal tensor $\mathcal{T}_1^{23}$

`Output:` list of matches $\mathcal{M}$, matching tables $\mathcal{J}_1$, $\mathcal{J}_2$, $\mathcal{J}_3$

Initialize $n=0$, $\mathcal{M}=\emptyset$, $\mathcal{J}_k(\mathbf{p})=0$ for all k, $\mathbf{p}$
Compute pairwise similarity scores $\mathbf{s}.s_{ab}$ and uniformity scores $\mathbf{s}.v$ for all seeds $\mathbf{s}$, $[\mathbf{s}.s_{ab}, \mathbf{s}.v] = \mathtt{sim}(\mathbf{s}, \mathcal{I}_{\mathbf{s}.a}, \mathcal{I}_{\mathbf{s}.b})$
Extend the two-view seeds to three views by trifocal transfer: $[\mathbf{s}.\mathbf{x}_c, \mathbf{s}.\mathbf{A}_{ac}] = \mathtt{transfer}(\mathbf{s}, \mathcal{T}_1^{23})$
Compute pairwise similarity scores $\mathbf{s}.s_{ac}$ for all seeds $\mathbf{s}$
Combine similarity scores, $\mathbf{s}.s = \mathtt{score}(\mathbf{s}.s_{ab}, \mathbf{s}.s_{ac}, z)$
Sort the seeds according to the scores $\mathbf{s}.s$
Initialize priority queue $\mathcal{Q}$ with sorted seeds
`while` $\mathcal{Q}$ not empty
  Draw the seed $\hat{\mathbf{q}} \in \mathcal{Q}$ with the best score $\hat{\mathbf{q}}.s$
  Set $a=\hat{\mathbf{q}}.a$ and $b=\hat{\mathbf{q}}.b$
  (In the following, $\mathcal{I}_a$ and $\mathcal{I}_b$ define new seeds)
  `for` each new match $\mathbf{q}^i$ nearby $\hat{\mathbf{q}}$ which satisfies the disparity gradient limit ϵ_d and the epipolar constraint ϵ_e
    Set $\mathbf{q}^i.s_{ab} = -\infty$
    `if` $\mathcal{J}_a(\mathtt{round}(\mathbf{q}^i.\mathbf{x}_a))=0$ `&` $\mathcal{J}_b(\mathtt{round}(\mathbf{q}^i.\mathbf{x}_b))=0$
      $[\mathbf{q}^i.s_{ab}, \mathbf{q}^i.v] = \mathtt{sim}(\mathbf{q}^i, \mathcal{I}_a, \mathcal{I}_b)$
  `end for`
  Sort matches $\mathbf{q}^i$ according to the scores $\mathbf{q}^i.s_{ab}$
  `for` each $\mathbf{q}^i$ satisfying $\mathbf{q}^i.s_{ab} \geq z$ and $\mathbf{q}^i.v \geq t$
    Set $n=n+1$
    `if` $\mathbf{q}^i.s_{ab} \geq z_\mathrm{u}$ and $\mathbf{q}^i.v \geq t_\mathrm{u}$
      Update $\mathbf{q}^i.\mathbf{A}_{ab}$, and thereafter $\mathbf{q}^i.a$ and $\mathbf{q}^i.b$
    Do trifocal transfer for $\mathbf{q}^i$ and compute $\mathbf{q}^i.s_{ac}$, $\mathbf{q}^i.s$
    `if` $\mathbf{q}^i.s_{ac} \geq \tilde{z}$ {
      Set $\mathcal{Q}=\mathcal{Q}\cup\{\mathbf{q}^i\}$ and $\mathcal{M}=\mathcal{M}\cup\{\mathbf{q}^i\}$
      Set $\mathcal{J}_a(\mathtt{round}(\mathbf{q}^i.\mathbf{x}_a))=n$, $\mathcal{J}_b(\mathtt{round}(\mathbf{q}^i.\mathbf{x}_b))=n$ }
    `if` $\mathcal{J}_c(\mathtt{round}(\mathbf{q}^i.\mathbf{x}_c))=0$ `&` $\mathbf{q}^i.s_{ac} \geq z$
      Set $\mathcal{J}_c(\mathtt{round}(\mathbf{q}^i.\mathbf{x}_c))=n$
  `end for`
`end while`

propagations. The scoring function is defined by

$$\texttt{score}(s_{ab}, s_{ac}, z) = \sum_{j \in \{b,c\}} \max\left(0, 1 - \frac{(s_{aj} - 1)^2}{(z-1)^2}\right), \tag{1.1}$$

which is a positive function on $[-1, 1]^3$. It is nonzero if either s_{ab} or s_{ac} exceed $z \leq 1$, and it obtains the largest values when they both exceed z and are close to 1.

Most of the computation time of Algorithm 2 is spent in the while loop. The first for loop is the same as in Algorithm 1. Thus, most of the bad candidate matches are rejected already on the basis of the pairwise score s_{ab} and the trifocal transfer and the evaluation of the total score are not necessary for them. In fact, only a fraction of the candidates survive to the second for loop, and hence, the three-view method is only slightly slower than the two-view method.

Finally, the last modifications are related to the acceptance of candidate matches and to the update of the matching tables (lines 22, 25, and 26 in Algorithm 2). Depending on the parameter settings, one may require that an accepted match must be visible in at least one or two pairs of views. That is, by setting $\tilde{z} = -1$, a candidate match is always accepted if its pairwise similarity score s_{ab} exceeds a threshold z and, on the other hand, setting $\tilde{z} = z$ implies that a match is accepted only if both s_{ab} and s_{ac} exceed z (line 22). However, in both cases, a new match is added to the matching table of the third view only if $s_{ac} \geq z$ and the corresponding pixel is not already reserved (lines 25 and 26).

3.3. *General multi-view matching*

As described in Algorithm 2, our current implementation of quasi-dense wide baseline matching can process at most three views simultaneously. However, this does not exclude applying the proposed approach to multi-view datasets with a large number of images, because there are several methods for combining multiple depth maps extracted from different view triplets [9; 37; 32]. Hence, Algorithm 2 could be used in generic multi-view stereo problems by first dividing large image sets into overlapping subsets of three images, then matching these image triplets and converting the obtained three-view matches to depth maps or point clouds, which are finally combined to a single model. In fact, decomposition into manageable subsets is indispensable for very large image sets but it may be advantageous also for smaller datasets in order to improve efficiency [42; 23; 13]. If the image subsets are processed in parallel and the resulting partial reconstructions are merged efficiently, it is possible to implement systems that reconstruct city-scale scenes from thousands or even millions of images within the span of a day on a single PC [13; 12].

Nevertheless, although there are multi-view reconstruction systems that build on two-view matching [51], in some cases it might be useful to be able to perform quasi-dense matching directly for a larger number of views than just two or three. This is the case with large unstructured photo collections since it may be practically

not feasible to process all possible view triplets but it is still not obvious how the optimal division into triplets should be done [13]. Hence, analogously to [14], it could be advantageous to extend Algorithm 2 for more than three views. In fact, given camera matrices, this may be relatively straightforward to do since it is easy to transfer the two-view matches to arbitrary number of views by the trifocal transfer (lines 3 and 21 in Algorithm 2) and define the score (1.1) as the sum of several pairwise terms. This would allow generalization to multiple views. However, our current implementation covers only view triplets and extension to more general settings could be seen as an interesting topic for future research.

4. Examples and experiments

We illustrate the performance of quasi-dense wide baseline matching in experiments with real images. The experiments in Sections 4.1 and 4.2 focus on two-view matching, Section 4.3 compares two-view and three-view approaches, and Sections 4.4 and 4.5 further illustrate three-view matching results.

4.1. *A simple example*

In our first experiment we demonstrate the two-view approach of Algorithm 1 by matching two views without using any additional prior knowledge, i.e., the fundamental matrix is not known a priori. However, in order to get quantitative accuracy estimates, we use sample views which are related by a homography and which were used in [38]. The example image pair is shown in Figure 1.1 and there is a significant change of scale between the views due to optical zooming. In addition, a seed match, i.e. a pair of elliptical regions, is also illustrated in Figure 1.1. The seed regions were extracted with the Hessian-Affine region detector [38] and automatically matched using the SIFT descriptor [35].

The match propagation was started from the single seed shown in Figure 1.1. The growing process resulted in 35589 matches and took 12 seconds on a 1GHz processor. Here we used only one seed in order to illustrate the fact that often already a one or a few correct matches are sufficient, but usually several tentative seed matches can be used since the algorithm is robust to outliers.

Given the grown matches, we fitted a homography to them by using a RANSAC-based estimation procedure. The resulting homography estimate was used to register the images on top of each other and the corresponding difference image is shown in Figure 1.1(b). Also, the grown quasi-dense matches are illustrated in Figure 1.1(e), where they are colored according to their homographic transfer error in the second image [20]. The majority of point correspondences have a displacement less than a pixel which indicates that the matches fit well to the estimated homography.

Interestingly, it also seems that our registration result is better than the homography estimate provided by the authors of [38], which is visualized in the last column of Figure 1.1. Indeed, the difference image in Figure 1.1(c) shows larger

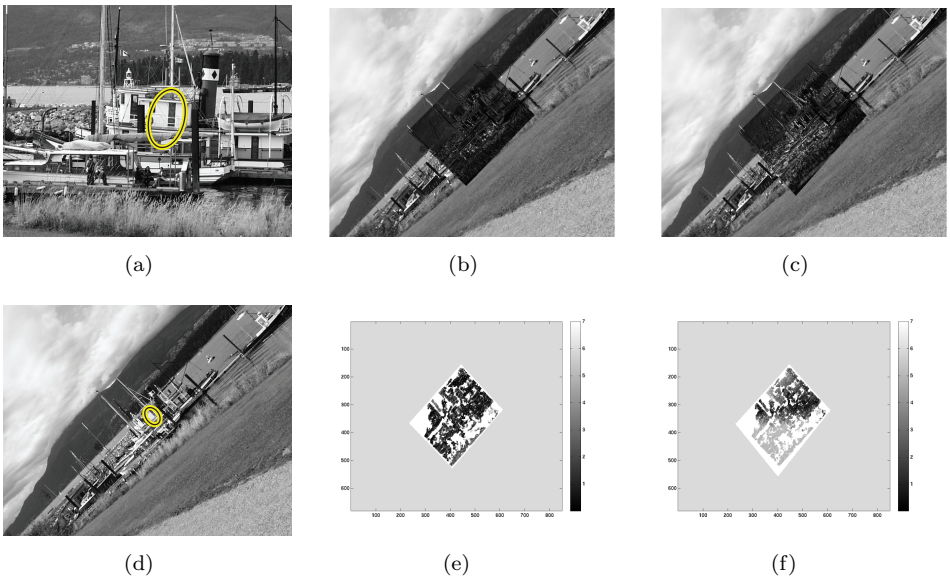
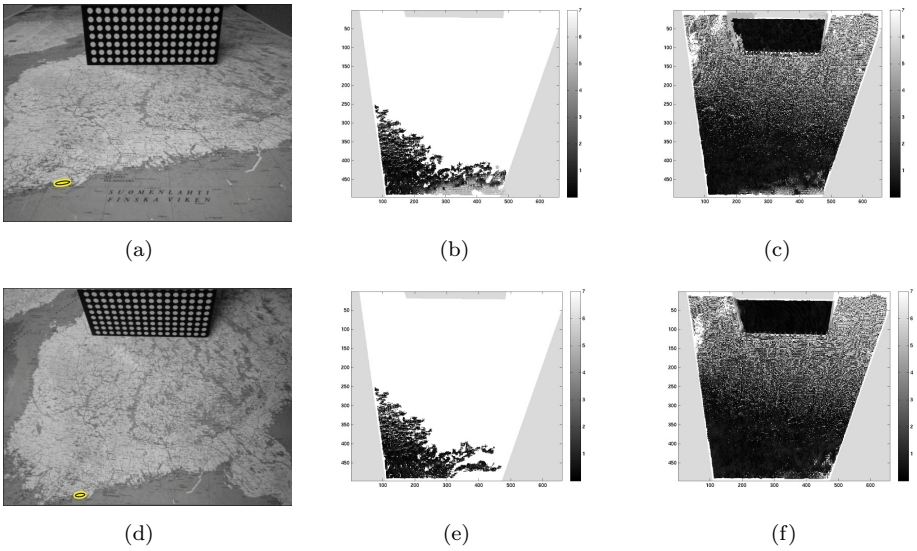


Fig. 1.1. Matching a pair of views which are related by a homography. (a) and (d): An elliptical region (a seed match) is shown on the original views, which are acquired by rotating and zooming a camera fixed on a tripod [38]. (b) and (c): The difference images obtained by aligning the views with homographies estimated by us and the authors of [38], respectively. (e) and (f): The quasi-dense matches grown from the single seed match are colored according to their distance from the location predicted by the two homographies, ours and theirs [38], respectively. (Image regions that are not visible in the first view have grayvalue 6 and the unmatched regions are white.)

intensity discrepancies than Figure 1.1(b). In addition, Figure 1.1(f) shows that the obtained quasi-dense matches are not consistent with the provided homography estimate in the lower left corner of the first image. This suggests that the homography estimate of [38] is not accurate there. Hence, we may note that even such a simple task as homography estimation is error-prone when registration landmarks are sparsely distributed and inaccurately localized. Moreover, our result shows that the quasi-dense approach can clearly improve image registration accuracy in such cases.

4.2. *An experiment with two views of a pair of planes*

The second experiment illustrates the performance of our two-view matching method under different parameter settings. The image pair used in this experiment is the same as in [24] and it is illustrated in Figures 1.2(a) and 1.2(d), which show two views of a scene containing two planes: a paper map on a table and a calibration plane orthogonal to the plane of the map. This pair of images is particularly suitable for algorithm evaluation since the calibration plane allows accurate estimation of the homographies which describe the mappings of the planes between



(a) (b) (c)

(d) (e) (f)

Fig. 1.2. Matching results for a pair of views of two planes using four different propagation settings. The left column shows the original images and one seed match (small ellipses in the lower left corner of the images). The last two columns illustrate the grown matches obtained by four parameter settings: (b) non-adaptive, (e) non-adaptive with epipolar constraint, (c) adaptive, and (f) adaptive with epipolar constraint. In each case, the matching pixels are colored according to their distance from the true location determined by the known homography of the respective plane. The values over 5 are suppressed to 5, and the non-common image area has grayvalue 6.

the two views, and hence, the obtained quasi-dence matches can be verified with the homographies after propagation. In addition, it can be seen that there is a clear perspective distortion between the views so that a global affine transformation is not a good approaximation for the homography of either plane. This allows to demonstrate the local affine adaptation capabilities of Algorithm 1.

The results obtained by Algorithm 1 using four different parameter settings are illustrated in Figure 1.2. In each case the match propagation was started from a single seed match, which is also shown in Figure 1.2 and which was extracted using the Hessian-Affine detector [38]. The pictures in the middle column of Figure 1.2 illustrate the grown quasi-dense matches obtained with the non-adaptive setting, i.e., without updating the affine transformation during propagation. The matches do not grow very far from the seed match because the initial affine transformation estimate is not accurate there due to perspective distortion. By comparing Figures 1.2(b) and 1.2(e), it can also be seen that imposing the epipolar constraint reduces the number of erroneous matches.

The adaptive match propagation results, obtained with and without epipolar constraint [24; 25], are shown in the last column of Figure 1.2, where the grown matches cover almost all the pixels in areas that are visible in both images. Again,

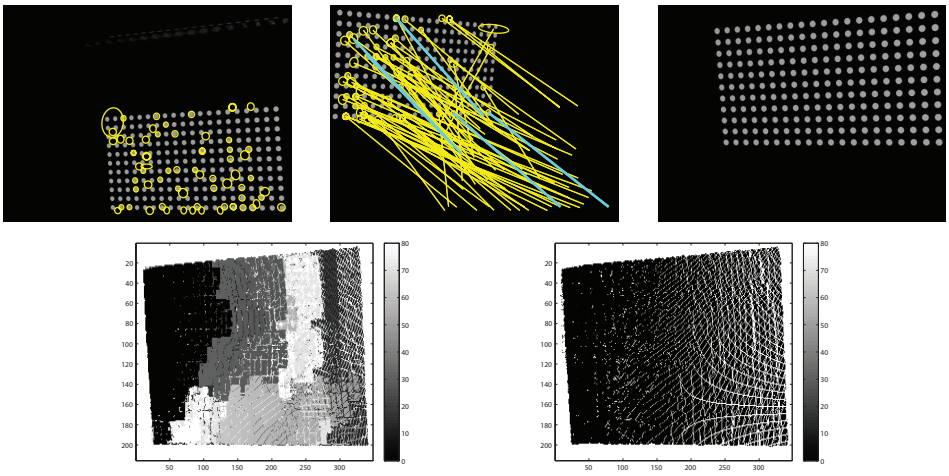


Fig. 1.3. Two-view and three-view matching of a plane with repetitive texture. Top: Three views and the elliptical seed regions extracted from the first pair of views. The flow vectors in the second view indicate the tentative seed region correspondences. There are only four correct seeds (cyan colored). Bottom: Disparity maps for the two-view (left) and three-view methods (right). Matched pixels are colored according to their distance to the correct corresponding location.

the epipolar constraint reduces false matches, as expected. The good coverage of matches shows that the update of affine transformation parameters on the basis of local texture properties has been successful and allows the single seed match to propagate into regions where the local geometric transformation between the views differs from the initial one. It is also interesting to note that the matching has expanded from the horizontal plane of the map to the vertical plane of the calibration object although there is a discontinuity in surface orientation. Further, the accuracy of grown matches does not decrease during propagation, i.e., majority of matched points have an error less than a pixel on both planes which indicates that errors do not accumulate and adaptation is stable.

4.3. *Matching three views of a plane with repetitive texture*

In the third experiment, we compare our two-view and three-view matching approaches by using several view triplets of a planar calibration pattern. As the sample views in Figure 1.3 show, the calibration pattern has a repetitive texture, which consists of a regular grid of white dots on a black background. Matching such a repetitive pattern is error-prone because the epipolar constraint alone is not sufficient to resolve the ambiguities in local matching. Thus, it is expected that three-view approach performs better than using just two views and our experiment aims to verify this hypothesis. In fact, because we know the true structure of the scene, it is easy to recover the correct homographies from manually picked correspondences and use them to assess the accuracy of obtained matches afterwards.

In detail, the experiment was carried out as follows. We selected a triplet of views, extracted seed matches from two of them [38], and performed both the two-view growing and the three-view growing with the same seeds. This was repeated for 100 triplets. Usually there were many incorrect seeds because the epipolar constraint was not effective in removing them due to repetitive texture. This is illustrated in Figure 1.3 where only four seeds are correct. However, also in this case, the three-view method (Algorithm 2) produced almost errorless matching whereas the two-view method (Algorithm 1) made many errors. This shows that the trifocal constraint allows to greatly improve the robustness of matching for repetitive textures. The results for all 100 triplets are shown in Table 1.1, where the total number of matches produced is approximately the same for both methods (i.e. 5.7 million). However, as the quartiles of the error distribution indicate, the matches produced by Algorithm 1 have a larger error. Thus, the two-view method failed in many cases whereas the three-view method was able to produce accurate matches for most of the cases. In this experiment the running time of Algorithm 2 was about 1.2 times the time of Algorithm 1. Hence, the three-view result is usually better than the result of two pairwise propagations and can be computed faster.

4.4. *Reconstruction of a piecewise planar scene*

Our fourth experiment compares the three-view matching method of Algorithm 2 with the multi-view stereo method of [14], which uses patch-based surface representation and relies on repeated patch expansion and filtering stages. The original implementation of [14] is available in source format and has produced good reconstruction results on publicly available benchmark datasets [46; 50].

As our current implementation can process at most three views simultaneously, we concentrated on comparing the basic correspondence growing processes on a dataset of three views. That is, we used our own images of a piecewise planar scene, whose structure can be accurately recovered from the images utilizing prior knowledge about the planes. However, this prior knowledge was only used to obtain the ground truth, i.e. a triangular mesh model of the scene, which was then used for accuracy evaluation of the point cloud reconstructions obtained by the two methods.

The images used in the experiment are illustrated in Figure 1.4 (top), and they show three planes, i.e., a calibration object consisting of two orthogonal planes on a wooden floor. There are regular patterns of circular dots on the planes which allowed to recover the pose of each plane as well as the camera matrices associated

Table 1.1. Experiment with repetitive texture. Quartiles of the error distribution are computed from 100 view triplets of the same kind as the triplet in Figure 1.3.

Method	No. points	1st quartile	Median	3rd quartile
Two views	5664285	0.33	0.74	11
Three views	5676160	0.079	0.27	0.65

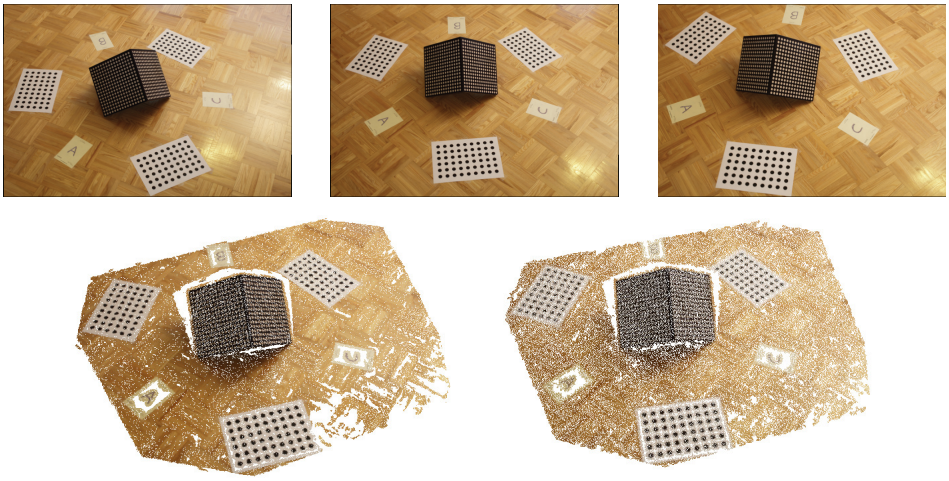


Fig. 1.4. Top: Three views of a piecewise planar scene. Left: Point cloud obtained by our three-view algorithm. Right: Point cloud computed by Furukawa's approach [14]. Accuracy of these two point clouds is illustrated by the curves with symbols $\times$ and $\circ$ in Figure 1.5, respectively.

to the three views. Given the views and the camera matrices, we reconstructed a point cloud using both Furukawa's approach and our approach (Algorithm 2). In the latter case, the seed matches were automatically extracted and matched using the Hessian-Affine detector [38] and SIFT descriptor [35], and after the correspondence growing by Algorithm 2 each grown match was triangulated from its two reference views (indexed by a and b in Algorithm 2).

The obtained point clouds are illustrated in Figure 1.4 (bottom). Our reconstruction consists of 155107 points and the match propagation stage took 228 seconds. Furukawa's point cloud has 119932 points, i.e. the centroids of reconstructed patches, and the processing time was 616 seconds, which also includes the time used for initial feature matching. However, as the running time of initial matching is negligible compared to that of patch expansion and filtering, it can be deduced that our correspondence growing is faster than Furukawa's. This is expected as we use only one expansion stage and we do not refine the alignment of matched patches via non-linear optimization during growing as in [14].

The accuracy of reconstructed points is evaluated by computing their distances to the ground truth surface, which is a triangular mesh representing the imaged planes. Further, each matched pixel in the three images corresponds to a point in the point cloud and the associated error is the distance of this point from the ground truth surface. Thus, each pixel is assigned with an error value, and we may assess reconstructions by comparing the error distributions of matched pixels. That is, we discretize the error values into 13 bins and compute the corresponding cumulative error histograms [50], which are visualized in Figure 1.5. By comparing the curves with symbols $\times$ and $\circ$, we may see that the accuracies of points in our

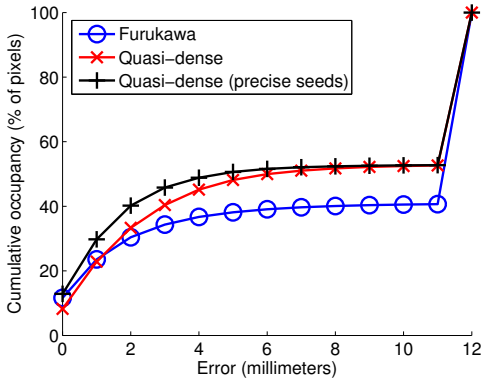


Fig. 1.5. Evaluation of accuracy for the point cloud reconstructions in Figure 1.4. Each matched pixel corresponds to a point in the point cloud and the associated error is the distance of this point from the ground truth surface. The curves illustrate the proportion of pixels whose error is less than a threshold. The curves are plotted using discrete error histograms of 13 bins where the last bin contains all pixels with an error > 11.5 mm, also the unmatched ones.

point cloud and Furukawa's point cloud are quite similar but our point cloud is denser. However, this is not very significant difference as the overall coverage of Furukawa's reconstruction is also good. The proportion of matched pixels is much less than 100 % in all cases because both methods used such parameter settings that required the reconstructed patches to be visible in all the three images. Hence, only those scene regions that are unoccluded in all three views could be reconstructed.

In addition, there is also a third curve in Figure 1.5, denoted by + symbol, and it illustrates the result obtained by our approach when all the circular dots of calibration patterns were used as seed regions (initialized with the correct pairwise affine transformations obtained from the ground truth planes). Although this is not a realistic application scenario for unknown scenes, the result indicates that it might be possible to further improve the accuracy of our approach by iteratively refining the seeds before growing them. In fact, this could be one direction for future developments as such a refinement of seeds would probably not decrease the overall computational efficiency very much since the number of seeds is usually small compared to the final number of grown matches.

4.5. *Examples with benchmark datasets*

In the last experiment, we compare our three-view matching method [29] and Furukawa's method [14] using benchmark datasets from [46]. The datasets contain images of two objects, called Dino and Temple. Three sample images of both objects are shown in Figures 1.6 and 1.7. There is also an online evaluation service [46] which allows to compare submitted reconstructions (i.e. mesh models) to ground-truth models of the objects obtained via a laser scanning process. However, since

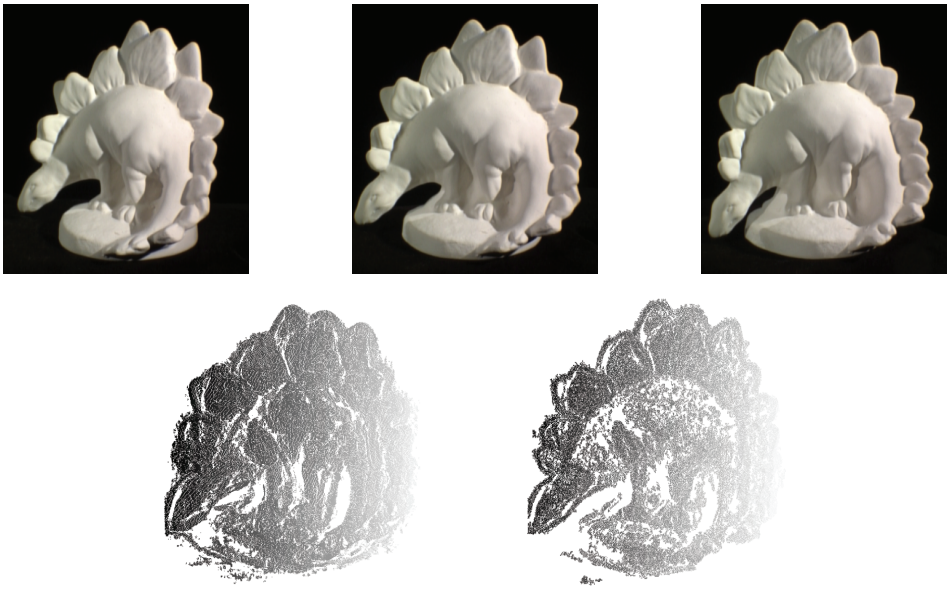


Fig. 1.6. Top: Three views from Dino dataset [46]. Bottom: Depth maps produced by our approach (left) and Furukawa's approach [14] (right). Note that the points in the depth maps are shown from the viewpoint of the second camera but they are colored according to their distance from the image plane of the first camera.

this chapter mainly presents a basic algorithm for match expansion in two-view and three-view cases and we do not discuss depth map fusion or mesh generation which would be both needed for generating complete object models, we confine ourselves to comparing three-view matching results qualitatively via visualized depth maps.

The experiment was carried out in a similar manner as the previous one in Section 4.4. The results are illustrated with depth maps in Figures 1.6 and 1.7 and they basically confirm the findings of the previous section. That is, our approach and Furukawa's approach provide comparable results but our approach is faster. For example, in the Dino example in Figure 1.6, the number of reconstructed points is 123633 in our point cloud and 90013 in Furukawa's point cloud. The correspondence growing stage of our approach took 442 seconds whereas Furukawa's approach took 621 seconds alltogether.

On the other hand, match expansion is implemented in different ways in the two methods [29; 14] and this may sometimes imply differences to the resulting point clouds. Thus, although the results are generally comparable there may be differences in some cases. For instance, as some parts of the object are not visible in all three views in Figure 1.7, we ran our algorithm using such parameters that the matching patches were not required to be visible in all views (i.e. we set in $\tilde{z} = -1$ in Algorithm 2). This resulted in slightly improved coverage of matched pixels but without significantly increasing the number of false matches, as shown

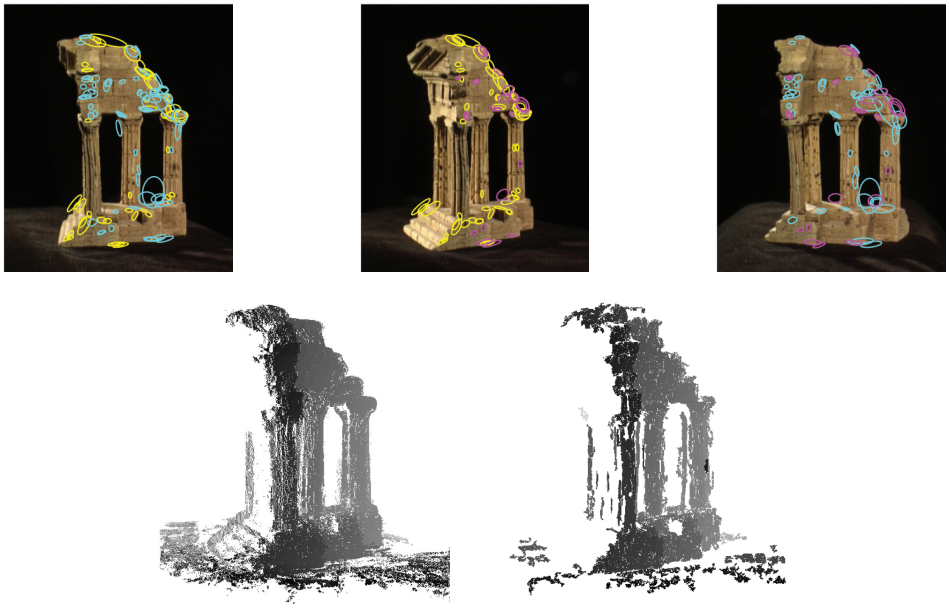


Fig. 1.7. Top: Three views from Temple dataset [46]. (The ellipses denote seed matches.) Bottom: Depth maps computed by our approach [29] (left) and Furukawa's approach [14] (right).

in Figure 1.7. However, in order to ensure the robustness of Furukawa's approach it was important to require that all reconstructed patches were visible in all three views. That is, we tried Furukawa's method also so that it accepted patches visible in only two images but this increased outliers and the result was worse than the one shown in Figure 1.7. This might be due to the fact that the best-first expansion strategy is not used in [14].

In summary, the results with Dino and Temple datasets suggest that accurate correspondence growing is possible without expensive iterative optimization for the pose of matched patches. Hence, although iterative surface refinement is probably necessary at the final stage for the best reconstruction results [14], it might be unnecessary at the correspondence growing stage.

5. Discussion

Besides describing the basic match propagation algorithms for quasi-dense wide baseline matching, this chapter has mainly concentrated on multi-view stereo reconstruction applications, where the algorithms are used for guided matching of calibrated images. That is, the camera matrices are assumed known, as is typical in multi-view stereo problems, and the fundamental matrix or trifocal tensor is used to guide image matching, i.e., to reject matches that do not satisfy multi-view constraints. The experiments of Section 4 demonstrate that our algorithms

are computationally efficient and provide results that are comparable to the state of the art [14]. In addition, as our current implementation can process only two or three views simultaneously, we outlined possibilities to extend the algorithms to more generic settings so that they would be directly applicable to arbitratry number of views. As discussed in Section 3.3, this seems to be a promising topic for future research.

However, besides multi-view reconstruction, the basic matching algorithms of Section 3 could be utilized in other applications as well. For example, unguided matching for image registration and multi-view geometry estimation, as illustrated in Section 4.1, is one evident application. In addition, quasi-dense matching can be used for registration of non-rigid deformations and for recognition and retrieval of particular objects [25; 4; 7]. Also, dense pixel-wise motion segmentation from multiple views may benefit from correspondence growing techniques [26].

6. Conclusion

In this chapter, we have described algorithms for quasi-dense wide baseline matching of view pairs and view triplets. The basic idea of the algorithms is to expand a sparse set of seed matches into a quasi-dense set of corresponding points between the views. The expansion is based on the best-first match propagation strategy. The algorithms can be used for unguided matching of view pairs of rigid and non-rigid scenes and for guided matching of view pairs and view triplets of rigid scenes. In the unguided case the quasi-dense approach allows to improve the robustness and accuracy of multi-view geometry estimation. On the other hand, multi-view surface reconstruction is the main application of guided matching, which refers to the case where matching is guided by the fundamental matrix or trifocal tensor. In this case, the algorithms can be used for depth map estimation from view pairs and triplets, and the obtained results are comparable to the state of the art within correspondence growing methods. Importantly, a good matching result is usually achieved with a single growth stage and without iterative refinement for the matched patches during growing. This is promising from the viewpoint of computational efficiency and encourages further studies on the topic.

References

1. Agarwal, S. *et al.* (2009). Building Rome in a day, in *ICCV*.
2. Boykov, Y., Veksler, O. and Zabih, R. (2001). Fast approximate energy minimization via graph cuts, *IEEE Trans. Pattern Anal. Mach. Intell.* **23**, 11, pp. 1222–1239.
3. Campbell, N. D. F., Vogiatzis, G., Hernández, C. and Cipolla, R. (2008). Using multiple hypotheses to improve depth-maps for multi-view stereo, in *ECCV*.
4. Cech, J., Matas, J. and Perdoch, M. (2010). Efficient sequential correspondence selection by cosegmentation, *IEEE Trans. Pattern Anal. Mach. Intell.* **32**, 9, pp. 1568–1581.

5. Cech, J. and Sára, R. (2007). Efficient sampling of disparity space for fast and accurate matching, in *CVPR*.
6. Chauve, A.-L., Labatut, P. and Pons, J.-P. (2010). Robust piecewise-planar 3d reconstruction and completion from large-scale unstructured point data, in *CVPR*.
7. Cho, M., Shin, Y. M. and Lee, K. M. (2010). Unsupervised detection and segmentation of identical objects, in *CVPR*.
8. Collins, R. T. (1996). A space-sweep approach to true multi-image matching, in *CVPR*.
9. Curless, B. and Levoy, M. (1996). A volumetric method for building complex models from range images, in *SIGGRAPH*.
10. Delaunoy, A. *et al.* (2008). Minimizing the multi-view stereo reprojection error for triangular surface meshes, in *BMVC*.
11. Faugeras, O. D. and Keriven, R. (1998). Variational principles, surface evolution, pdes, level set methods, and the stereo problem, *IEEE Transactions on Image Processing* **7**, 3, pp. 336–344.
12. Frahm, J.-M. *et al.* (2010). Building Rome on a cloudless day, in *ECCV*.
13. Furukawa, Y., Curless, B., Seitz, S. M. and Szeliski, R. (2010). Towards Internet-scale multi-view stereo, in *CVPR*.
14. Furukawa, Y. and Ponce, J. (2010). Accurate, dense, and robust multiview stereopsis, *IEEE Trans. Pattern Anal. Mach. Intell.* **32**, 8, pp. 1362–1376.
15. Gallup, D. *et al.* (2007). Real-time plane-sweeping stereo with multiple sweeping directions, in *CVPR*.
16. Gargallo, P. (2008). *Contributions to the Bayesian approach to multi-view stereo*, Ph.D. thesis, Institut National Polytechnique de Grenoble.
17. Goesele, M. *et al.* (2007). Multi-view stereo for community photo collections, in *ICCV*.
18. Grün, A. (1985). Adaptive least squares correlation: a powerful image matching technique, *S. Afr. J. of Photogrammetry, Remote Sensing and Cartography* **14**, 3.
19. Habbecke, M. and Kobbelt, L. (2007). A surface-growing approach to multi-view stereo reconstruction, in *CVPR*.
20. Hartley, R. and Zisserman, A. (2000). *Multiple View Geometry in Computer Vision* (Cambridge).
21. Hiep, V. H., Keriven, R., Labatut, P. and Pons, J.-P. (2009). Towards high-resolution large-scale multi-view stereo, in *CVPR*.
22. Hirschmüller, H. (2008). Stereo processing by semiglobal matching and mutual information, *IEEE Trans. Pattern Anal. Mach. Intell.* **30**, 2, pp. 328–341.
23. Jancosek, M., Shekhovtsov, A. and Pajdla, T. (2009). Scalable multi-view stereo, in *3DIM*.
24. Kannala, J. and Brandt, S. S. (2007). Quasi-dense wide baseline matching using match propagation, in *CVPR*.
25. Kannala, J., Rahtu, E., Brandt, S. S. and Heikkilä, J. (2008). Object recognition and segmentation by non-rigid quasi-dense matching, in *CVPR*.
26. Kannala, J., Rahtu, E., Brandt, S. S. and Heikkilä, J. (2009). Dense and deformable motion segmentation for wide baseline images, in *SCIA*.
27. Kobbelt, L. and Botsch, M. (2004). A survey of point-based techniques in computer graphics, *Computers & Graphics* **28**, 6, pp. 801–814.
28. Kolev, K., Klodt, M., Brox, T. and Cremers, D. (2009). Continuous global optimization in multiview 3d reconstruction, *Int. J. Comput. Vis.* **84**, 1, pp. 80–96.
29. Koskenkorva, P., Kannala, J. and Brandt, S. S. (2010). Quasi-dense wide baseline matching for three views, in *ICPR*.
30. Kutulakos, K. N. and Seitz, S. M. (2000). A theory of shape by space carving, *Int. J. Comput. Vis.* **38**, 3, pp. 199–218.

31. Labatut, P., Keriven, R. and Pons, J.-P. (2007). Efficient multi-view reconstruction of large-scale scenes using interest points, Delaunay triangulation and graph cuts, in *ICCV*.
32. Labatut, P., Pons, J.-P. and Keriven, R. (2009). Robust and efficient surface reconstruction from range data, *Comput. Graph. Forum* **28**, 8, pp. 2275–2290.
33. Lhuillier, M. and Quan, L. (2002). Match propagation for image-based modeling and rendering, *IEEE Trans. Pattern Anal. Mach. Intell.* **24**, 8.
34. Liu, Y., Dai, Q. and Xu, W. (2010). A point-cloud-based multiview stereo algorithm for free-viewpoint video, *IEEE Trans. Vis. Comput. Graph.* **16**, 3, pp. 407–418.
35. Lowe, D. (2004). Distinctive image features from scale invariant keypoints, *Int. J. Comput. Vis.* **60**, pp. 91–110.
36. Martinec, D. (2008). *Robust multiview reconstruction*, Ph.D. thesis, Czech Technical University.
37. Merrell, P. *et al.* (2007). Real-time visibility-based fusion of depth maps, in *ICCV*.
38. Mikolajczyk, K. *et al.* (2005). A comparison of affine region detectors, *Int. J. Comput. Vis.* **65**, pp. 43–72.
39. Mouragnon, E. *et al.* (2009). Generic and real-time structure from motion using local bundle adjustment, *Image and Vision Computing* **27**, pp. 1178–1193.
40. Newcombe, R. A. and Davison, A. J. (2010). Live dense reconstruction with a single moving camera, in *CVPR*.
41. Otto, G. P. and Chau, T. K. W. (1989). "Region-growing" algorithm for matching of terrain images, *Image Vision Comput.* **7**, 2, pp. 83–94.
42. Pollefeys, M. *et al.* (2008). Detailed real-time urban 3D reconstruction from video, *Int. J. Comput. Vis.* **78**, 2-3, pp. 143–167.
43. Scharstein, D. and Szeliski, R. (2002). A taxonomy and evaluation of dense two-frame stereo correspondence algorithms, *Int. J. Comput. Vis.* **47**, 1-3, pp. 7–42.
44. Seitz, S. M. and Dyer, C. R. (1999). Photorealistic scene reconstruction by voxel coloring, *Int. J. Comput. Vis.* **35**, 2, pp. 151–173.
45. Seitz, S. M. *et al.* (2006a). A comparison and evaluation of multi-view stereo reconstruction algorithms, in *CVPR*.
46. Seitz, S. M. *et al.* (2006b). Datasets for evaluation of multi-view stereo reconstruction algorithms, *http://vision.middlebury.edu/mview/* .
47. Sinha, S. N., Mordohai, P. and Pollefeys, M. (2007). Multi-view stereo via graph cuts on the dual of an adaptive tetrahedral mesh, in *ICCV*.
48. Snavely, N., Seitz, S. M. and Szeliski, R. (2008). Modeling the world from internet photo collections, *Int. J. Comput. Vis. (IJCV)* **80**, 2, pp. 189–210.
49. Strecha, C., Tuytelaars, T. and Gool, L. J. V. (2003). Dense matching of multiple wide-baseline views, in *ICCV*.
50. Strecha, C. *et al.* (2008). On benchmarking camera calibration and multi-view stereo for high resolution imagery, in *CVPR*.
51. Tylecek, R. and Sara, R. (2010). Refinement of surface mesh for accurate multi-view reconstruction, *International Journal of Virtual Reality* **9**, 1, pp. 45–54.
52. Xiao, J., Chen, J., Yeung, D.-Y. and Quan, L. (2008). Learning two-view stereo matching, in *ECCV*.