Accurate, Dense, and Robust Multi-View Stereopsis

Yasutaka Furukawa[1]
Department of Computer Science
and Beckman Institute
University of Illinois at Urbana-Champaign, USA[1]

Jean Ponce[1,2]
Willow Team–ENS/INRIA/ENPC
Département d'Informatique
Ecole Normale Supérieure, Paris, France[2]

Abstract: *This paper proposes a novel algorithm for calibrated multi-view stereopsis that outputs a (quasi) dense set of rectangular patches covering the surfaces visible in the input images. This algorithm does not require any initialization in the form of a bounding volume, and it detects and discards automatically outliers and obstacles. It does not perform any smoothing across nearby features, yet is currently the top performer in terms of both coverage and accuracy for four of the six benchmark datasets presented in [20]. The keys to its performance are effective techniques for enforcing local photometric consistency and global visibility constraints. Stereopsis is implemented as a* match, expand, and filter *procedure, starting from a sparse set of matched keypoints, and repeatedly expanding these to nearby pixel correspondences before using visibility constraints to filter away false matches. A simple but effective method for turning the resulting patch model into a mesh appropriate for image-based modeling is also presented. The proposed approach is demonstrated on various datasets including objects with fine surface details, deep concavities, and thin structures, outdoor scenes observed from a restricted set of viewpoints, and "crowded" scenes where moving obstacles appear in different places in multiple images of a static structure of interest.*

1. Introduction

As in the binocular case, although most early work in multi-view stereopsis (e.g., [12, 15, 19]) tended to match and reconstruct all scene points independently, recent approaches typically cast this problem as a variational one, where the objective is to find the surface minimizing a global photometric discrepancy functional, regularized by explicit smoothness constraints [1, 8, 17, 18, 22, 23] (a geometric consistency terms is sometimes added as well [3, 4, 7, 9]). Competing approaches mostly differ in the type of optimization techniques that they use, ranging from local methods such as gradient descent [3, 4, 7], level sets [1, 9, 18], or expectation maximization [21], to global ones such as graph cuts [3, 8, 17, 22, 23]. The variational approach has led to impressive progress, and several of the methods recently surveyed by Seitz et al. [20] achieve a relative accuracy better than 1/200 (1mm for a 20cm wide object) from a set of low-resolution (640×480) images. However, it typically requires determining a bounding volume (valid depth range, bounding box, or visual hull) prior to initiating the optimization process, which may not be feasible for outdoor scenes and/or cluttered images. [1] We propose instead a simple and efficient algorithm for calibrated multi-view stereopsis that does not require any initialization, is capable of detecting and discarding outliers and obstacles, and outputs a (quasi) dense collection of small oriented rectangular patches [6, 13], obtained from pixel-level correspondences and tightly covering the observed surfaces except in small textureless or occluded regions. It does not perform any smoothing across nearby features, yet is currently the top performer in terms of both coverage and accuracy for four of the six benchmark datasets provided in [20]. The keys to its performance are effective techniques for enforcing local photometric consistency and global visibility constraints. Stereopsis is implemented as a *match, expand, and filter* procedure, starting from a sparse set of matched keypoints, and repeatedly expanding these to nearby pixel correspondences before using visibility constraints to filter away false matches. A simple but effective method for turning the resulting patch model into a mesh suitable for image-based modeling is also presented. The proposed approach is applied to three classes of datasets:

- *objects*, where a single, compact object is usually fully visible in a set of uncluttered images taken from all around it, and it is relatively straightforward to extract the apparent contours of the object and compute its visual hull;
- *scenes*, where the target object(s) may be partially occluded and/or embedded in clutter, and the range of viewpoints may be severely limited, preventing the computation of effective bounding volumes (typical examples are outdoor scenes with buildings or walls); and

[1] In addition, variational approaches typically involve massive optimization tasks with tens of thousands of coupled variables, potentially limiting the resolution of the corresponding reconstructions (see, however, [18] for a fast GPU implementation). We will revisit tradeoffs between computational efficiency and reconstruction accuracy in Sect. 5.

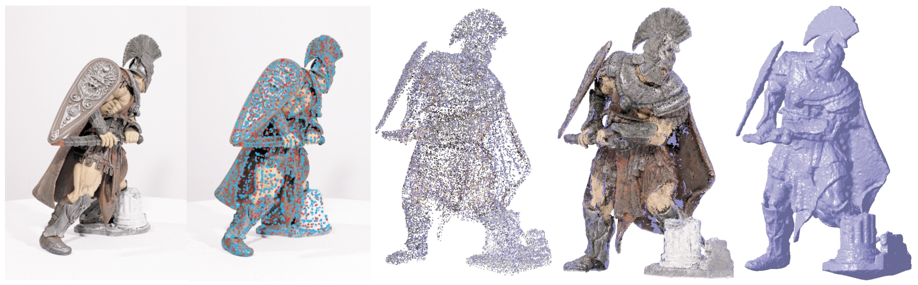


Figure 1. Overall approach. From left to right: a sample input image; detected features; reconstructed patches after the initial matching; final patches after expansion and filtering; polygonal surface extracted from reconstructed patches.

• *crowded scenes*, where moving obstacles appear in different places in multiple images of a static structure of interest (e.g., people passing in front of a building).

Techniques such as space carving [12, 15, 19] and variational methods based on gradient descent [3, 4, 7], level sets [1, 9, 18], or graph cuts [3, 8, 17, 22, 23] typically require an initial bounding volume and/or a wide range of viewpoints. Object datasets are the ideal input for these algorithms, but methods using multiple depth maps [5, 21] or small, independent surface elements [6, 13] are better suited to the more challenging scene datasets. Crowded scenes are even more difficult. The method proposed in [21] uses expectation maximization and multiple depth maps to reconstruct a crowded scene despite the presence of occluders, but it is limited to a small number of images (typically three). As shown by qualitative and quantitative experiments in the rest of this paper, our algorithm effectively handles all three types of data, and, in particular, outputs accurate object and scene models with fine surface detail despite low-texture regions, large concavities, and/or thin, high-curvature parts. As noted earlier, it implements multi-view stereopsis as a simple *match, expand, and filter* procedure (Fig. 1): (1) *Matching*: features found by Harris and Difference-of-Gaussians operators are matched across multiple pictures, yielding a sparse set of patches associated with salient image regions. Given these initial matches, the following two steps are repeated n times ($n = 3$ in all our experiments): (2) *Expansion:* a technique similar to [16, 2, 11, 13] is used to spread the initial matches to nearby pixels and obtain a dense set of patches. (3) *Filtering:* visibility constraints are used to eliminate incorrect matches lying either in front or behind the observed surface. This approach is similar to the method proposed by Lhuillier and Quan [13], but their expansion procedure is greedy, while our algorithm iterates between expansion and filtering steps, which allows us to process complicated surfaces. Furthermore, outliers cannot be handled in their method. These differences are also true with the approach by Kushal and Ponce [11] in comparison to ours. In addition, only a pair of images can be handled at once in [11], while our method can process arbitrary number of images uniformly.

2. Key Elements of the Proposed Approach

Before detailing our algorithm in Sect. 3, we define here the patches that will make up our reconstructions, as well as the data structures used throughout to represent the input images. We also introduce two other fundamental building blocks of our approach, namely, the methods used to accurately reconstruct a patch once the corresponding image fragments have been matched, and determine its visibility.

2.1. Patch Models

A patch p is a rectangle with center $c(p)$ and unit normal vector $n(p)$ oriented toward the cameras observing it (Fig. 2). We associate with p a reference image $R(p)$, chosen so that its retinal plane is close to parallel to p with little distortion. In turn, $R(p)$ determines the orientation and extent of the rectangle p in the plane orthogonal to $n(p)$, so the projection of one of its edges into $R(p)$ is parallel to the image rows, and the smallest axis-aligned square containing its image covers a $\mu \times \mu$pixel^2 area (we use values of 5 or 7 for μ in all of our experiments). Two sets of pictures are also attached to each patch p: the images $S(p)$ where p *should* be visible (despite self-occlusion), but may in practice not be recognizable (due to highlights, motion blur, etc.), or hidden by moving obstacles, and the images $T(p)$ where it is *truly* found ($R(p)$ is of course an element of $T(p)$). We enforce the following two constraints on the model: First, we enforce local *photometric consistency* by requiring that the projected textures of every patch p be consistent in at least γ images (in other words $|T(p)| \geq \gamma$, with $\gamma = 3$ in all but three of our experiments, where γ is set to 2). Second, we enforce global *visibility consistency* by requiring that no patch p be occluded by any other patch in any image in $S(p)$. [2]

2.2. Image Models

We associate with each image I a regular grid of $\beta_1 \times \beta_1$pixel^2 cells $C(i, j)$, and attempt to reconstruct at least one

[2]A patch p *may* be occluded in one or several of the images in $S(p)$ by moving obstacles, but these are *not* reconstructed by our algorithm and thus do not generate occluding patches.

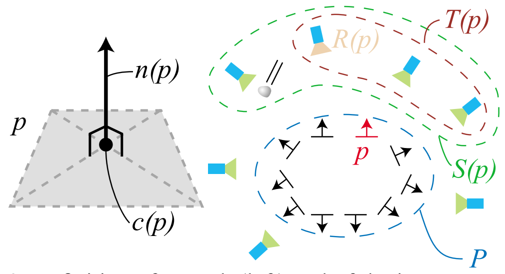


Figure 2. Definition of a patch (left) and of the images associated with it (right). See text for the details.

patch in every cell (we use values of 1 or 2 for β_1 in all our experiments). The cell $C(i,j)$ keeps track of two different sets $Q_t(i,j)$ and $Q_f(i,j)$ of reconstructed patches potentially visible in $C(i,j)$: A patch p is stored in $Q_t(i,j)$ if $I \in T(p)$, and in $Q_f(i,j)$ if $I \in S(p) \setminus T(p)$. We also associate with $C(i,j)$ the depth of the center of the patch in $Q_t(i,j)$ closest to the optical center of the corresponding camera. This amounts to attaching a depth map to I, which will prove useful in the visibility calculations of Sect. 2.4.

2.3. Enforcing Photometric Consistency

Given a patch p, we use the normalized cross correlation (NCC) $N(p,I,J)$ of its projections into the images I and J to measure their photometric consistency. Concretely, a $\mu \times \mu$ grid is overlaid on p and projected into the two images, the correlated values being obtained through bilinear interpolation. Given a patch p, its reference image $R(p)$, and the set of images $T(p)$ where it is truly visible, we can now estimate its position $c(p)$ and its surface normal $n(p)$ by maximizing the average NCC score

$$\bar{N}(p) = \frac{1}{|T(p)|-1} \sum_{I \in T(p), I \neq R(p)} N(p,R(p),I) \qquad (1)$$

with respect to these unknowns. To simplify computations, we constrain $c(p)$ to lie on the ray joining the optical center of the reference camera to the corresponding image point, reducing the number of degrees of freedom of this optimization problem to three—depth along the ray plus yaw and pitch angles for $n(p)$, and use a conjugate gradient method [14] to find the optimal parameters. Simple methods for computing reasonable initial guesses for $c(p)$ and $n(p)$ are given in Sects. 3.1 and 3.2.

2.4. Enforcing Visibility Consistency

The visibility of each patch p is determined by the images $S(p)$ and $T(p)$ where it is (potentially or truly) observed. We use two slightly different methods for constructing $S(p)$ and $T(p)$ depending on the stage of our reconstruction algorithm. In the matching phase (Sect. 3.1), patches are reconstructed from sparse feature matches, and we have to rely on photometric consistency constraints to determine (or rather obtain an initial guess for) visibility. Concretely, we initialize both sets of images as those for which the NCC score exceeds some threshold: $S(p) = T(p) = \{I | N(p,R(p),I) > \alpha_0\}$. On the other hand, in the expansion phase of our algorithm (Sect. 3.2), patches are by construction dense enough to associate depth maps with all images, and $S(p)$ is constructed for each patch by thresholding these depth maps—that is, $S(p) = \{I | d_I(p) \leq d_I(i,j) + \rho_1\}$, where $d_I(p)$ is the depth of the center of p along the corresponding ray of image I, and $d_I(i,j)$ is the depth recorded in the cell $C(i,j)$ associated with image I and patch p. The value of ρ_1 is determined automatically as the distance at the depth of $c(p)$ corresponding to an image displacement of β_1 pixels in $R(p)$. Once $S(p)$ has been estimated, photometric consistency is used to determine the images where p is truly observed as $T(p) = \{I \in S(p) | N(p,R(p),I) > \alpha_1\}$. This process may fail when the reference image $R(p)$ is itself an outlier, but, as explained in the next section, our algorithm is designed to handle this problem. Iterating its matching and filtering steps also helps improve the reliability and consistency of the visibility information.

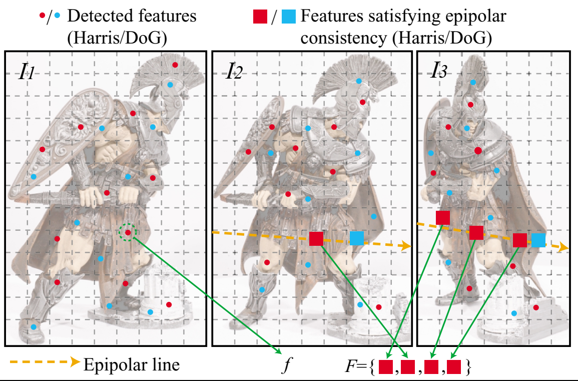
3. Algorithm

3.1. Matching

As the first step of our algorithm, we detect corner and blob features in each image using the Harris and Difference-of-Gaussian (DoG) operators. [3] To ensure uniform coverage, we lay over each image a coarse regular grid of $\beta_2 \times \beta_2$pixel^2 cells, and return as corners and blobs for each cell the η local maxima of the two operators with strongest responses (we use $\beta_2 = 32$ and $\eta = 4$ in all our experiments). After these features have been found in each image, they are matched across multiple pictures to reconstruct a sparse set of patches, which are then stored in the grid of cells $C(i,j)$ overlaid on each image (Fig. 3): Consider an image I and denote by O the optical center of the corresponding camera. For each feature f detected in I, we collect in the other images the set F of features f' of the same type (Harris or DoG) that lie within $\iota = 2$pixels from the corresponding epipolar lines, and triangulate the 3D points associated with the pairs (f,f'). We then consider these points in order of increasing distance from O as potential patch centers, [4] and return the first patch "photoconsistent" in at least γ images (Fig. 3, top). More concretely, for each

[3]Briefly, let us denote by G_σ a 2D Gaussian with standard deviation σ. The response of the Harris filter at some image point is defined as $H = \det(M) - \lambda \mathrm{trace}^2(M)$, where $M = G_{\sigma_0} * (\nabla I \nabla I^T)$, and ∇I is computed by convolving the image I with the partial derivatives of the Gaussian G_{σ_1}. The response of the DoG filter is $D = |(G_{\sigma_2} - G_{\sqrt{2}\sigma_2}) * I|$. We use $\sigma_0 = \sigma_1 = \sigma_2 = 1$pixel and $\lambda = 0.06$ in all of our experiments.

[4]Empirically, this heuristic has proven to be effective in selecting mostly correct matches at a modest computational expense.


●/● Detected features (Harris/DoG)
🟥/🟦 Features satisfying epipolar consistency (Harris/DoG)
I_1
I_2
I_3
Epipolar line
f
F={🟥,🟥,🟥,🟥}

```
Input: Features detected in each image.
Output: Initial sparse set of patches P.

Cover each image with a grid of $\beta_1 \times \beta_1$pixel$^2$ cells;
$P \leftarrow \phi$;
For each image $I$ with optical center $O$
  For each feature $f$ detected in $I$ and lying in an empty cell
    $F \leftarrow$ {Features satisfying the epipolar consistency};
    Sort $F$ in an increasing order of distance from $O$;
    For each feature $f' \in F$
      $R(p) \leftarrow I$; $T(p) \leftarrow \{J|N(p,R(p),J) \geq \alpha_0\}$;
      $c(p) \leftarrow$ 3D point triangulated from $f$ and $f'$;
      $n(p) \leftarrow$ Direction of optical ray from $c(p)$ to $O$;
      $n(p), c(p) \leftarrow$ argmax $\bar{N}(p)$;
      $S(p) \leftarrow \{J|N(p,R(p),J) \geq \alpha_0\}$;
      $T(p) \leftarrow \{J|N(p,R(p),J) \geq \alpha_1\}$;
      If $|T(p)| \geq \gamma$
        register $p$ to the corresponding cells in $S(p)$;
        exit innermost For loop, and add $p$ to $P$.
```

Figure 3. Feature matching algorithm. Top: An example showing the features $f' \in F$ satisfying the epipolar constraint in images I_2 and I_3 as they are matched to feature f in image I_1 (this is an illustration only, not showing actual detected features). Bottom: The matching algorithm. The values used for α_0 and α_1 in all our experiments are 0.4 and 0.7 respectively.

feature f', we construct the potential surface patch p by triangulating f and f' to obtain an estimate of $c(p)$, assign to $n(p)$ the direction of the optical ray joining this point to O, and set $R(p) = I$. After initializing $T(p)$ by using photometric consistency as in Sect. 2.4, we use the optimization process described in Sect. 2.3 to refine the parameters of $c(p)$ and $n(p)$, then initialize $S(p)$ and recompute $T(p)$. Finally, if p satisfies the constraint $|T(p)| \geq \gamma$, we compute its projections in all images in $S(p)$, register it to the corresponding cells, and add it to P (Fig. 3, bottom). Note that since the purpose of this step is only to reconstruct an initial, sparse set of patches, features lying in non-empty cells are skipped for efficiency. Also note that the patch generation process may fail if the reference image $R(p)$ is an outlier, for example when f correspond to a highlight. This does not prevent, however, the reconstruction of the corresponding surface patch from another image. The second part of our algorithm iterates (three times in all our experiments) between an expansion step to obtain dense patches and a filtering step to remove erroneous matches and enforce visibility consistency, as detailed in the next two sections.

3.2. Expansion

At this stage, we iteratively add new neighbors to existing patches until they cover the surfaces visible in the scene. Intuitively, two patches p and p' are considered to be neighbors when they are stored in adjacent cells $C(i,j)$ and $C(i',j')$ of the same image I in $S(p)$, and their tangent planes are close to each other. We only attempt to create new neighbors when necessary—that is, when $Q_t(i',j')$ is empty,[5] and none of the elements of $Q_f(i',j')$ is *n-adjacent* to p, where two patches p and p' are said to be n-adjacent when $|(c(p)-c(p'))\cdot n(p)|+|(c(p)-c(p'))\cdot n(p')| < 2\rho_2$. Similar to ρ_1, ρ_2 is determined automatically as the distance at the depth of the mid-point of $c(p)$ and $c(p')$ corresponding to an image displacement of β_1 pixels in $R(p)$. When these two conditions are verified, we initialize the patch p' by assigning to $R(p')$, $T(p')$, and $n(p')$ the corresponding values for p, and assigning to $c(p')$ the point where the viewing ray passing through the center of $C(i',j')$ intersects the plane containing the patch p. Next, $c(p')$ and $n(p')$ are refined by the optimization procedure discussed in Sect. 2.3, and $S(p')$ is initialized from the depth maps as explained in Sect. 2.4. Since some matches (and thus the corresponding depth map information) may be incorrect at this point, the elements of $T(p')$ are added to $S(p')$ to avoid missing any image where p' may be visible. Finally, after updating $T(p')$ using photometric constraints as in Sect. 2.4, we accept the patch p' if $|T(p')| \geq \gamma$ still holds, then register it to $Q_t(i',j')$ and $Q_f(i',j')$, and update the depth maps associated with images in $S(p')$. See Fig. 4 for the algorithm.

3.3. Filtering

Two filtering steps are applied to the reconstructed patches to further enforce visibility consistency and remove erroneous matches. The first filter focuses on removing patches that lie outside the real surface (Fig. 5, left): Consider a patch p_0 and denote by U the set of patches that it occludes. We remove p_0 as an outlier when $|T(p_0)|\bar{N}(p_0) < \sum_{p_j \in U} \bar{N}(p_j)$ (intuitively, when p_0 is an outlier, both $\bar{N}(p_0)$ and $|T(p_0)|$ are expected to be small, and p_0 is likely to be removed). The second filter focuses on outliers lying inside the actual surface (Fig. 5, right): We simply recompute $S(p_0)$ and $T(p_0)$ for each patch p_0 using the depth maps associated with the corresponding images (Sect. 2.4), and

[5]Intuitively, any patch p' in $Q_t(i',j')$ would either already be a neighbor of p, or be separated from it by a depth discontinuity, neither case warranting the addition of a new neighbor.

```
Input:  Patches $P$ from the feature matching step.
Output:  Expanded set of reconstructed patches.

Use $P$ to initialize, for each image, $Q_f$, $Q_t$, and its depth map.
While $P$ is not empty
  Pick and remove a patch $p$ from $P$;
  For each image $I \in T(p)$ and cell $C(i,j)$ that $p$ projects onto
    For each cell $C(i',j')$ adjacent to $C(i,j)$ such that $Q_t(i',j')$
    is empty and $p$ is not n-adjacent to any patch in $Q_f(i',j')$
      Create a new $p'$, copying $R(p'), T(p')$ and $n(p')$ from $p$;
      $c(p') \leftarrow$ Intersection of optical ray through
                    center of $C(i',j')$ with plane of $p$;
      $n(p'), c(p') \leftarrow \mathrm{argmax} \bar{N}(p')$;
      $S(p') \leftarrow$ {Visible images of $p'$ estimated by the
                   current depth maps } $\cup\, T(p')$;
      $T(p') \leftarrow \{J \in S(p') | N(p', R(p'), J) \geq \alpha_1\}$;
      If $|T(p') < \gamma|$, go back to For-loop;
      Add $p'$ to $P$;
      Update $Q_t, Q_f$ and depth maps for $S(p')$;
Return all the reconstructed patches stored in $Q_f$ and $Q_t$.
```

Figure 4. Patch expansion algorithm.

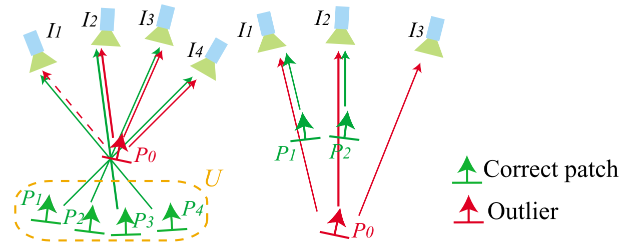


Figure 5. Outliers lying outside (left) or inside (right) the correct surface. Arrows are drawn between the patches p_i and the images I_j in $S(p_i)$, while solid arrows correspond to the case where $I_j \in T(p_i)$. U denotes a set of patches occluded by an outlier. See text for details.

remove p_0 when $|T(p_0)| < \gamma$. Note that the recomputed values of $S(p_0)$ and $T(p_0)$ may be different from those obtained in the expansion step since more patches have been computed after the reconstruction of p_0. Finally, we enforce a weak form of regularization as follows: For each patch p, we collect the patches lying in its own and adjacent cells in all images of $S(p)$. If the proportion of patches that are *n-adjacent* to p in this set is lower than $\epsilon = 0.25$, p is removed as an outlier. The threshold α_1 is initialized with 0.7, and lowered by 0.2 after each expansion/filtering iteration.

4. Polygonal Surface Reconstruction

The reconstructed patches form an *oriented point*, or *surfel* model. Despite the growing popularity of this type of models in the computer graphics community [10], it remains desirable to turn our collection of patches into surface meshes for image-based modeling applications. The approach that we have adopted is a variant of the iterative deformation algorithm presented in [4], and consists of two phases. Briefly, after initializing a polygonal surface from a predetermined bounding volume, the convex hull of the reconstructed points, or a set of small hemispheres centered at these points and pointing away from the cameras, we repeatedly move each vertex v according to three forces (Fig. 6): a *smoothness* term for regularization; a *photometric consistency* term, which is based on the reconstructed patches in the first phase, but is computed solely from the mesh in the second phase; and, when accurate silhouettes are available, a *rim consistency* term pulling the rim of the deforming surface toward the corresponding visual cones.

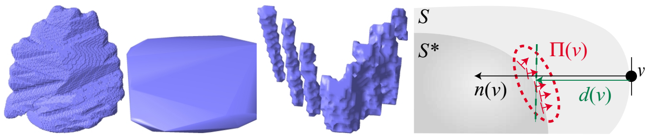


Figure 6. Polygonal surface reconstruction. Left: bounding volumes for the *dino* (visual hull), *steps* (convex hull), and *city-hall* (union of hemispheres) datasets featured in Figs. 7 ,9 and 10. Right: geometric elements driving the deformation process.

Concretely, the smoothness term is $-\zeta_1 \Delta v + \zeta_2 \Delta^2 v$, where Δ denotes the (discrete) Laplacian operator relative to a local parameterization of the tangent plane in v ($\zeta_1 = 0.6$ and $\zeta_2 = 0.4$ are used in all our experiments). In the first phase, the photometric consistency term for each vertex v essentially drives the surface towards reconstructed patches and is given by $\nu(v) n(v)$, where $n(v)$ is the inward unit normal to S in v, $\nu(v) = \max(-\tau, \min(\tau, d(v)))$, and $d(v)$ is the signed distance between v and the true surface S^* along $n(v)$ (the parameter τ is used to bound the magnitude of the force, ensure stable deformation and avoid self-intersections; its value is fixed as 0.2 times the average edge length in S). In turn, $d(v)$ is estimated as follows: We collect the set $\Pi(v)$ of $\pi = 10$ patches p' with (outward) normals compatible with that of v (that is, $-n(p') \cdot n(v) > 0$, see Fig. 6) that lie closest to the line defined by v and $n(v)$, and compute $d(v)$ as the weighted average distance from v to the centers of the patches in $\Pi(v)$ along $n(v)$—that is, $d(v) = \sum_{p' \in \Pi(v)} w(p')[n(v) \cdot (c(p') - v)]$, where the weights $w(p')$ are Gaussian functions of the distance between $c(p')$ and the line, with standard deviation ρ_1 defined as before, and normalized to sum to 1. In the second phase, the photometric consistency term is computed for each vertex by using the patch optimization routine as follows. At each vertex v, we create a patch p by initializing $c(p)$ with v, $n(p)$ with a surface normal estimated at v on S, and a set of visible images $S(p)$ from a depth-map testing on the mesh S at v, then apply the patch optimization routine described in Sect. 2.3. Let $c^*(p)$ denote the value of $c(p)$ after the optimization, then $c^*(p) - c(p)$ is used as the photometric consistency term. In the first phase, we iterate until conver-

Table 1. Characteristics of the datasets used in our experiments. *roman* and *skull* datasets have been acquired in our lab, while other datasets have been kindly provided by S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski (*temple* and *dino*, see also [20]); C. Hernández Esteban, F. Schmitt and the Museum of Cherbourg (*polynesian*); S. Sullivan and Industrial Light and Magic (*face*, *face-2*, *body*, *steps*, and *wall*); and C. Strecha (*city-hall* and *brussels*).

Name	Images	Image Size	β_1	μ	γ
roman	48	1800×1200	1	5	3
temple	16	640×480	1	5	3
dino	16	640×480	1	7	3
skull	24	2000×2000	2	5	3
polynesian	36	1700×2100	2	5	3
face	4	1400×2200	1	7	2
face-2	13	1500×1400	1	5	3
body	4	1400×2200	1	7	2
steps	7	1500×1400	1	7	3
city-hall	5	3000×2000	2	7	3
wall	9	1500×1400	1	7	3
brussels	3	2000×1300	1	5	2

gence, remesh, increase the resolution of the surface, and repeat the process until the desired resolution is obtained (in particular, until image projections of edges of the mesh become approximately β_1 pixels in length, see [4] for details). The second phase is applied to the mesh only in its desired resolution as a final refinement.

5. Experiments and Discussion

We have implemented the proposed approach in C++, using the WNLIB [14] implementation of conjugate gradient in the patch optimization routine. The datasets used in our experiments are listed in Table 1, together with the number of input images, their approximate size and a choice of parameters for each data set. Note that all the parameters except for β_1, μ and γ have been fixed in our experiments.

We have first tested our algorithm on *object* datasets (Figs. 1 and 7) for which a segmentation mask is available in each image. A visual hull model is thus used to initialize the iterative deformation process for all these datasets, except for *face* and *body*, where a limited set of viewpoints is available, and the convex hull of the reconstructed patches is used instead. The segmentation mask is also used by our stereo algorithm, which simply ignores the background during feature detection and matching. The rim consistency term has only been used in the surface deformation process for the *roman* and *skull* datasets, for which accurate contours are available. The bounding volume information has *not* been used to filter out erroneous matches in our experiments. Our algorithm has successfully reconstructed various surface structures such as the high-curvature and/or shallow surface details of *roman*, the thin cheek bone and deep eye sockets of *skull*, and the intricate facial features of *face* and *face-2*. Quantitative comparisons kindly provided by D. Scharstein on the datasets presented in [20] show that the proposed method outperforms all the other evaluated techniques in terms of *accuracy* (distance d such that a given percentage of the reconstruction is within d from the ground truth model) and *completeness* (percentage of the ground truth model that is within a given distance from the reconstruction) on four out of the six datasets. The datasets consists of two objects (*temple* and *dino*), each of which constitutes three datasets (*sparse ring, ring,* and *full*) with different numbers of input images, ranging from 16 to more than 300, and our method achieves the best accuracy and completeness on all the *dino* datasets and the smallest *sparse ring temple*. Note that the *sparse ring temple* and *dino* datasets consisting of 16 views have been shown in Fig. 7 and their quantitative comparison with the top performers [4, 5, 7, 18, 21, 22, 23] are given in Fig. 8. [6] Finally, the bottom part of Fig. 8 compares our algorithm with Hernández Esteban's method [7], which is one of the best multi-view stereo reconstruction algorithms today, for the *polynesian* dataset, where a laser scanned model is used as a ground truth. As shown by the close-ups in this figure, our model is qualitatively better than the Herández's model, especially at sharp concave structures. This is also shown quantitatively using the same accuracy and completeness measures as before.

Reconstruction results for *scene* datasets are shown in Fig. 9. Additional information (such as segmentation masks, bounding boxes, or valid depth ranges) is not available in this case. The *city-hall* example is interesting because viewpoints change significantly across input cameras, and part of the building is only visible in some of the frames. Nonetheless, our algorithm has successfully reconstructed the whole scene with fine structural details. The *wall* dataset is challenging since a large portion of several of the input pictures consists of running water, and the corresponding image regions have successfully been detected as outliers, while accurate surface details have been recovered for the rigid wall structure. Finally, Fig. 10 illustrates our results on *crowded scene* datasets. Our algorithm reconstructs the background building from the *brussels* dataset, despite people occluding various parts of the scene. The *steps-2* dataset is an artificially generated example, where we have manually painted a red cartoonish human in each image of *steps* images. To further test the robustness of our algorithm against outliers, the *steps-3* dataset has been created from *steps-2* by copying its images but replacing the fifth one with the third, without changing camera parameters. This is a particularly challenging example, since the whole fifth image must be detected as an outlier. We have successfully reconstructed the details of both despite these outliers. Note

[6] Rendered views of the reconstructions and all the quantitative evaluations can be found at `http://vision.middlebury.edu/mview/`.

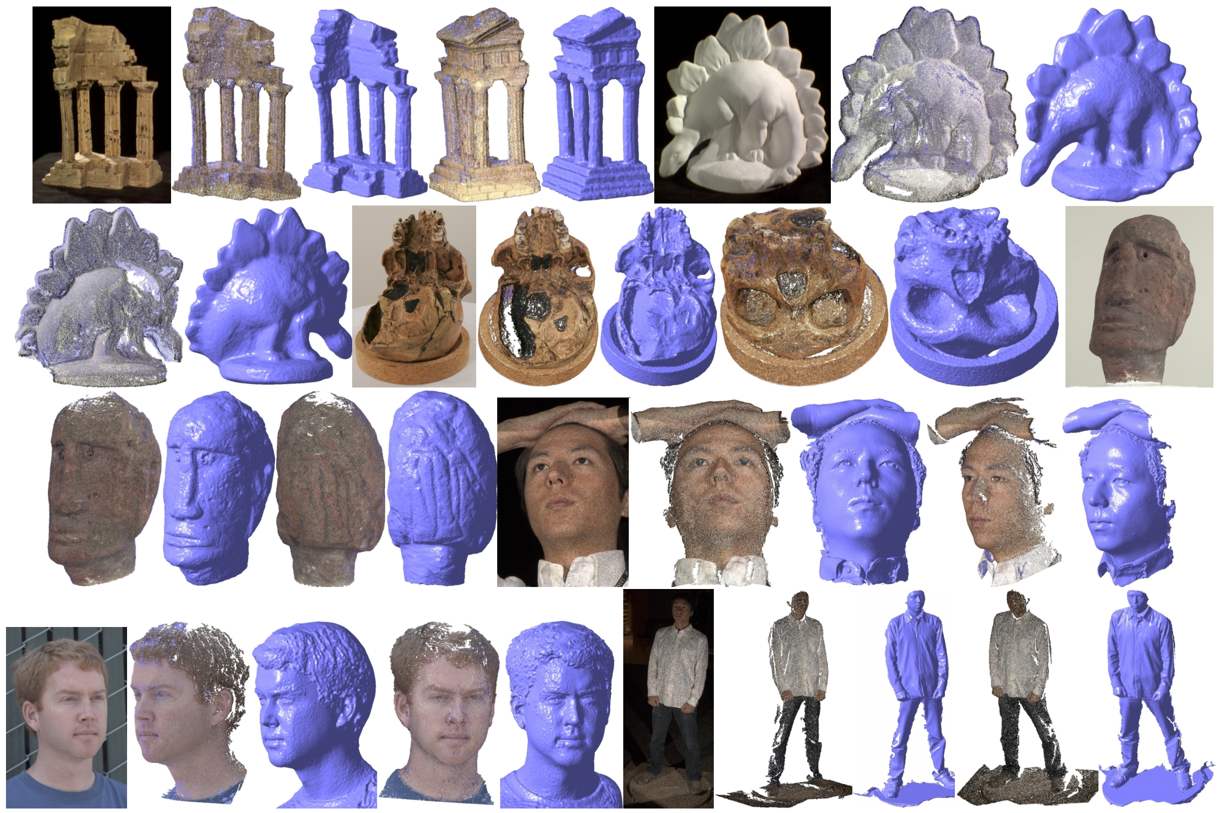


Figure 7. Sample results on object datasets: From left to right and top to bottom: *temple*, *dino*, *skull*, *polynesian*, *face*, *face-2*, and *body* datasets. In each case, one of the input image is shown, along with two views of texture-mapped reconstructed patches and shaded polygonal surfaces.

that the convex hull of the reconstructed patches' centers is used for the surface initialization except for the *city-hall* and *brussels*, for which the union of hemispheres is used.

The bottleneck of our multi-view stereo matching algorithm is the patch expansion step, whose running time varies from about 20 minutes, for small datasets such as *temple* and *dino*, to up to a few hours for datasets consisting of high-resolution images, such as *polynesian* and *city-hall*. The running times of polygonal surface extraction also range from 30 minutes to a few hours depending on the size of datasets. This is comparable to many variational methods [20], despite the fact that our algorithm does not involve any large optimization problem. This is due to several factors: First, unlike algorithms using voxels or discretized depth labels, our method solves a fully continuous optimization problem, thus does not suffer from discretization errors and can handle high-resolution input images directly, but trades speed for accuracy. Second, we use a region-based photometric consistency measure, which is much slower than a point-based measure, but takes into account surface orientation during optimization. In turn, this allows our algorithm to handle gracefully outdoor images with varying illumination. Again, accuracy and speed are conflicting requirements. To conclude, let us note that our future work will be aimed at adding a temporal component to our reconstruction algorithm, with the aim of achieving markerless face and body motion capture.

Acknowledgments: This work was supported in part by the National Science Foundation under grant IIS-0535152. We thank S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski for the *temple* and *dino* datasets and evaluations, C. Hernández Esteban, F. Schmitt, and the Museum of Cherbourg for *polynesian*, S. Sullivan, A. Suter, and Industrial Light and Magic for *face*, *face-2*, *body*, *steps*, and *wall*, C. Strecha for *city-hall* and *brussels*, and finally J. Blumenfeld and S. R. Leigh for the *skull* data set.

References

[1] O. Faugeras and R. Keriven. Variational principles, surface evolution, PDE's, level set methods and the stereo problem. *IEEE Trans. Im. Proc.*, 7(3):336–344, 1998. 1, 2

[2] V. Ferrari, T. Tuytelaars, and L. Van Gool. Simultaneous object recognition and segmentation by image exploration. In *ECCV*, 2004. 2

[3] Y. Furukawa and J. Ponce. Carved visual hulls for image-based modeling. In *ECCV*, volume 1, 2006. 1, 2

[4] Y. Furukawa and J. Ponce. High-fidelity image-based modeling. Technical Report CVR 2006-02, University of Illinois at Urbana-Champaign, 2006. 1, 2, 5, 6

[5] M. Goesele, B. Curless, and S. M. Seitz. Multi-view stereo revisited. In *CVPR*, pages 2402–2409, 2006. 2, 6

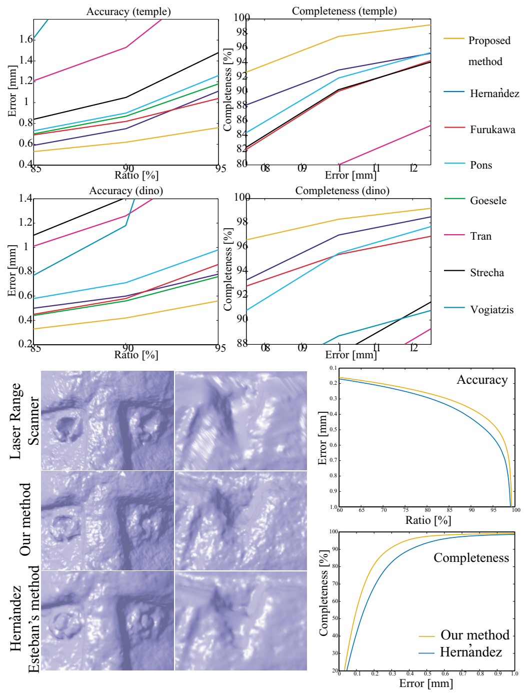


Figure 8. Quantitative comparison with other multi-view stereo algorithms for the *temple* and *dino* at the top, and for the *polynesian* at the bottom.

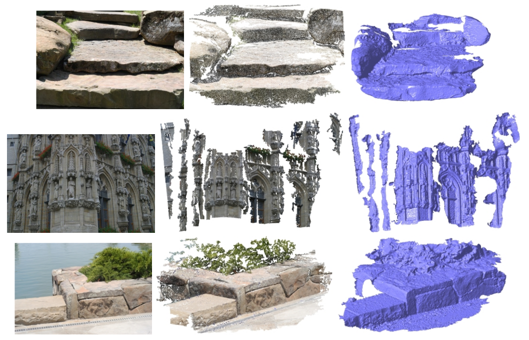


Figure 9. Sample results on *scene* datasets. From top to bottom: *steps*, *city-hall*, and *wall* datasets.

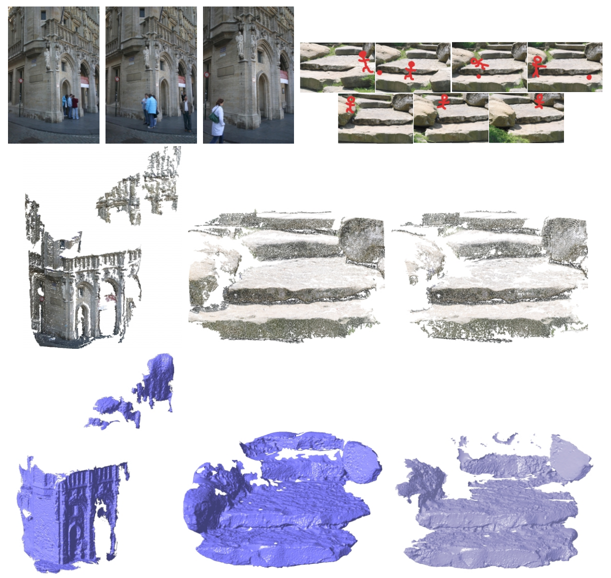


Figure 10. Sample results on *crowded scene* datasets. Top row: input images for *brussels* and *steps-2*; Second and third rows: reconstruction results for *brussels*, *steps-2*, and *steps-3*. Note that the *steps-3* dataset is generated by copying *steps-2* but replacing its third image by the fifth without changing camera parameters.

[6] M. Habbecke and L. Kobbelt. Iterative multi-view plane fitting. In *11th Fall Workshop on VISION, MODELING, AND VISUALIZATION*, 2006. 1, 2

[7] C. Hernández Esteban and F. Schmitt. Silhouette and stereo fusion for 3D object modeling. *CVIU*, 96(3), 2004. 1, 2, 6

[8] A. Hornung and L. Kobbelt. Hierarchical volumetric multi-view stereo reconstruction of manifold surfaces based on dual graph embedding. In *CVPR*, 2006. 1, 2

[9] R. Keriven. A variational framework to shape from contours. Technical Report 2002-221, ENPC, 2002. 1, 2

[10] L. Kobbelt and M. Botsch. A survey of point-based techniques in computer graphics. *Computers & Graphics*, 28(6):801–814, 2004. 5

[11] A. Kushal and J. Ponce. A novel approach to modeling 3d objects from stereo views and recognizing them in photographs. In *ECCV*, volume 2, pages 563–574, 2006. 2

[12] K. Kutulakos and S. Seitz. A theory of shape by space carving. *IJCV*, 38(3):199–218, 2000. 1, 2

[13] M. Lhuillier and L. Quan. A quasi-dense approach to surface reconstruction from uncalibrated images. *PAMI*, 27(3):418–433, 2005. 1, 2

[14] W. Naylor and B. Chapman. Wnlib. 3, 6

[15] M. Okutami and T. Kanade. A multiple-baseline stereo system. *PAMI*, 15(4):353–363, 1993. 1, 2

[16] G. P. Otto and T. K. W. Chau. 'region-growing' algorithm for matching of terrain images. *Image Vision Comput.*, 7(2):83–94, 1989. 2

[17] S. Paris, F. Sillion, and L. Quan. A surface reconstruction method using global graph cut optimization. In *ACCV*, January 2004. 1, 2

[18] J.-P. Pons, R. Keriven, and O. D. Faugeras. Modelling dynamic scenes by registering multi-view image sequences. In *CVPR (2)*, pages 822–827, 2005. 1, 2, 6

[19] S. Seitz and C. Dyer. Photorealistic scene reconstruction by voxel coloring. In *CVPR*, pages 1067–1073, 1997. 1, 2

[20] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. *CVPR*, 1, 2006. 1, 6, 7

[21] C. Strecha, R. Fransens, and L. V. Gool. Combined depth and outlier estimation in multi-view stereo. In *CVPR*, pages 2394–2401, 2006. 1, 2, 6

[22] S. Tran and L. Davis. 3d surface reconstruction using graph cuts with surface constraints. In *ECCV*, 2006. 1, 2, 6

[23] G. Vogiatzis, P. H. Torr, and R. Cipolla. Multi-view stereo via volumetric graph-cuts. In *CVPR*, 2005. 1, 2, 6