Eurographics Symposium on Geometry Processing 2010
Bruno Levy and Olga Sorkine
(Guest Editors)

Volume 29 (2010), Number 5

Globally Consistent Space-Time Reconstruction

T. Popa[1] I. South-Dickinson[2] D. Bradley[2] A. Sheffer[2] W. Heidrich[2]

1) ETH Zurich, 2) University of British Columbia

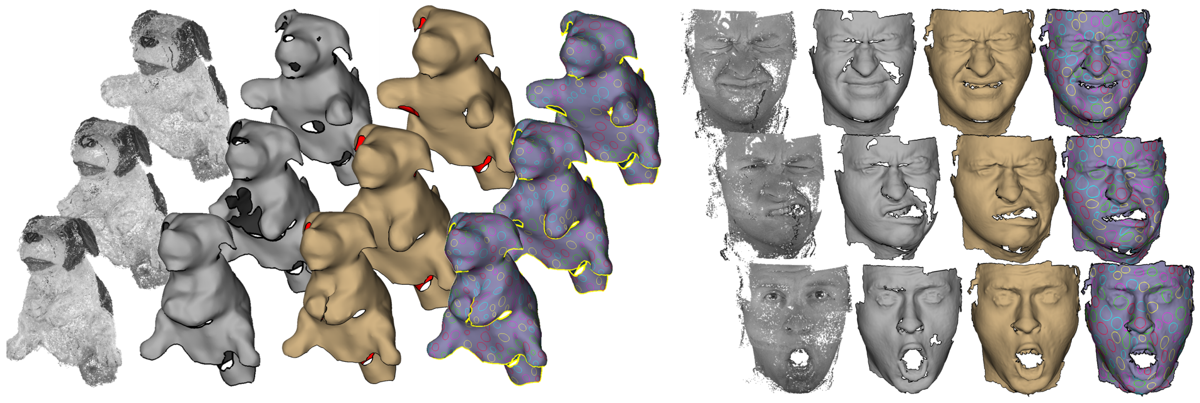

Figure 1: *Left to right: textured point clouds, individually reconstructed frames (note the gluing when the puppet's arms are folded or the actor's mouth closes), globally consistent reconstruction, circle pattern visualizes frame-to-frame correspondence.*

Abstract

Most objects deform gradually over time, without abrupt changes in geometry or topology, such as changes in genus. Correct space-time reconstruction of such objects should satisfy this gradual change *prior. This requirement necessitates a globally consistent interpretation of spatial adjacency. Consider the capture of a surface that comes in contact with itself during the deformation process, such as a hand with different fingers touching one another in parts of the sequence. Naive reconstruction would glue the contact regions together for the duration of each contact and keep them apart in other parts of the sequence. However such reconstruction violates the gradual change prior as it enforces a drastic intrinsic change in the object's geometry at the transition between the glued and unglued sub-sequences. Instead consistent global reconstruction should keep the surfaces separate throughout the entire sequence. We introduce a new method for globally consistent space-time geometry and motion reconstruction from video capture. We use the gradual change prior to resolve inconsistencies and faithfully reconstruct the geometry and motion of the scanned objects. In contrast to most previous methods our algorithm doesn't require a strong shape prior such as a template and provides better results than other template-free approaches.*

1. Introduction

Geometry scanning and capture techniques in combination with surface reconstruction methods provide a powerful and convenient way to create virtual replicas of real-world objects. Advanced scanning methods are now capable of providing raw capture data, typically point clouds, for dynamically deforming shapes. Reconstruction methods from such space-time data have three main goals [WAO*09, LAGP09]: preserving the captured per-frame geometry while filtering-out capture noise; computing the most likely object motion or frame-to-frame geometry correspondence; and completion of geometric information that is missing in some frames by using the computed motion and the data available in other frames. The latter two tasks must leverage knowledge of the temporal behavior of the captured objects and thus are

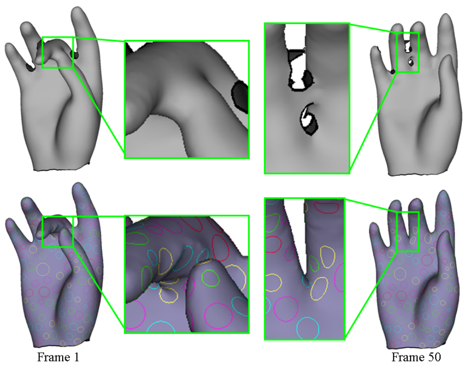

Figure 2: *Two frames of the hand sequence: (top) per-frame reconstruction; (bottom) our globally consistent reconstruction - the circle pattern visualizes the inter-frame correspondences.*

strongly linked to the motion priors assumed by the reconstruction method.

Most commonly scanned deforming objects change their shape gradually in terms of both Euclidean coordinates and intrinsic surface shape. This includes articulated objects, humans, animals, garments, and many other objects in our everyday surrounding. This *gradual change* observation implies that no abrupt changes are possible that would drastically affect the intrinsic shape, such as a change in the object's genus. Note that the gradual change prior does not prevent fairly large changes in the shape over longer periods of time. Correct, or *globally consistent* space-time reconstruction of such shapes should satisfy this prior.

One of the challenges in spatio-temporal reconstruction is to reconcile *inconsistent* per-frame inputs. Inconsistent in this context means that straightforward, naive combination of the per-frame scans violates the shape or motion priors. A common source of inconsistencies in captures of gradually changing shapes is self-contact situations, where the moving surface comes in contact with itself, such as when the dog's paws are folded, an actor's mouth closes (Figure 1), or the fingers of a human hand touch (Figure 2). Such self-contacts occur quite frequently in typical motions. Without additional priors, temporally local reconstruction of the contact subsequences would incorrectly glue the contact areas together (Figure 1, 2nd column, and Figure 2, top). However, when the complete sequence is available, globally consistent reconstruction should keep them apart (Figure 1, 3rd column, and Figure 2, bottom). Inconsistencies can also happen because of capture noise and errors in local reconstruction.

A template or another strong shape prior can help with resolving self-contacts or other inconsistencies [VBMP08, LAGP09, BHPS10]. However, the use of such a prior can be limiting, as creating templates for each particular capture requires a significant effort and possibly additional scanning hardware. Instead, Wand et al. [WJH*07, WAO*09] resolve inconsistencies by using the input data to construct a template-like surface representation and then deform it to approximate the inputs. The approximation can often lead to loss of local geometric details (Figure 3).

We present an alternative template-free, globally-consistent reconstruction method which uses a bottom-up approach to achieve better local detail preservation and which can handle more complex examples than those shown in [WJH*07, WAO*09]. We first reconstruct each frame independently, capturing local geometric details, using off-the shelf static reconstruction methods, and then assemble these frames into a globally consistent space-time frame sequence, computing the object's motion and completing missing information across time. The assembly process detects and corrects the inconsistencies present in the local reconstructions.

We observe that for many captures the gradual change prior by itself does not define a unique reconstruction result (consider for instance a sphere rotating around itself). Since our method focuses on reconstruction from video, we use the visual data, and specifically optical flow [Bou99], as an aid to reconstruct the most likely motion.

The key component of our method is computation of correspondence between consecutive frames in the sequence. We observe that since the change is gradual, the correspondence, or mapping between correctly reconstructed consecutive frames should exhibit very little stretch, as the intrinsic surface distances change very little with a small change in time. However, for the initial independently reconstructed frames such a mapping may not exist a priori, since due to occlusions and inconsistencies the frames might have different genus and other large intrinsic differences. To overcome these differences we developed a specialized mapping mechanism that uses a local to global approach, assembling a global mapping from a set of local ones, resulting in a map which allows for easy detection of inconsistencies and hole completion and which provides a low-stretch feasible correspondence in the consistent regions.

As demonstrated by the results (Section 7), our method can handle a variety of contact situations correctly, as it reconstructs the scanned shapes and accumulates both geometric details and motion information across the frame sequences.

2. Related Work

In recent years numerous methods for space-time reconstruction have been proposed, such as [HAWG08, TBW*09, PG08, SAL*08, GILM07, WAO*09, LAGP09, BHPS10, SWG08, FP09, dAST*08, ZST*10]. Most of these methods target gradually changing geometries, often making additional assumptions on the processed models.

Despite the frequency of surface self-contacts, few of the methods explicitly address contact scenarios or show examples where contacts occur [WJH*07, WAO*09, LAGP09, BHPS10, FP09, VZBH08]. Methods that aim to reconstruct both geometry and motion but ignore the possibility of contacts or inconsistencies, e.g. [BPS*08], can either provide unnatural results or fail completely when those are present. Methods that compute only the motion or frame-to-frame registration of the scanned object, [ATR*08, VZBH08,

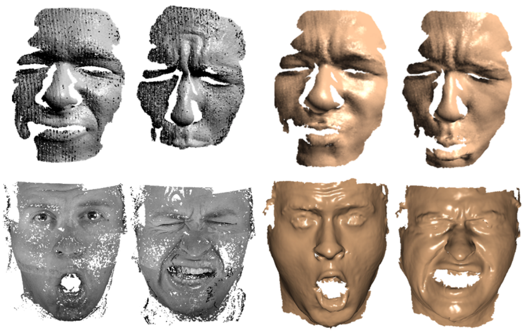

Figure 3: *Preserving local details: (top) results from [WAO*09]: point clouds (left) reconstruction (right). Note the mismatch in the wrinkles on the forehead. Bottom - facial reconstruction with our method (typical frames): textured point clouds [BHPS10] (left) reconstruction (right).*

MFO*07, HAWG08], can often overcome such inconsistencies. However without corresponding geometry reconstruction and data completion, their results are not as useful.

One approach for resolving inconsistencies is to introduce a strong shape prior and ignore local data not consistent with it. A few methods use an external template, typically aligned manually or semi-manually with the first frame [LAGP09, VBMP08]. Creating a template may require a significant user effort. Others use one of the frames in the sequence as a template, e.g. [SWG08, FP08, FP09, BHPS10]. This often involves manual cleanup of the selected frame, including data completion and noise removal. Finding a frame with good local reconstruction, such that the completion and correction required are minimal, can be a challenge for some sequences. Replacing the template with a skeleton [PH08], still requires significant user input and is only applicable to a specific subset of models.

A template-free, global approach for consistent reconstruction was proposed in [WJH*07] and improved upon in [WAO*09]. Their methods use the data to construct a template-like surface representation and then deform it to approximate the inputs. The approximation often smoothes out details or introduces non-existent details coming from the approximating surface (see Figure 3). Like these approaches our method does not rely on a template or other external priors. However, by adopting a bottom-up reconstruction approach combined with the new mapping mechanism it obtains better reconstruction results.

3. Algorithm Overview

The goal of our method is to reconstruct a *consistent frame sequence* from a given space-time point cloud using precomputed optical flow correspondences as an aid, robustly handling self-contacts and other local inconsistencies. We define a consistent frame sequence as a sequence of meshes with the same connectivity and with per-frame vertex positions that satisfy the gradual change assumptions of small spatial and intrinsic changes in the mesh. The shared connectivity explicitly defines the object's motion over time.

Initialization: When performing the reconstruction, we aim to maximize the use of per-frame geometric information.To this end, our method starts by constructing geometrically accurate per-frame meshes by performing noise and outlier removal on the point cloud. The obtained per-frame mesh geometry is preserved as much as possible throughout the rest of the pipeline. We found the method of [BBH08] for reconstruction from passive video to be well suited for this task.

Hierarchical Assembly: To assemble the reconstructed per-frame meshes into a single globally consistent sequence we use a hierarchical sequence assembly procedure that, at each step, combines pairs of consecutive consistent frame sub-sequences into a single consistent sequence (Figure 4). At level zero (left column), we pair individual frames. At level one, we pair sub-sequences of length two, and so forth. The assembly mechanism can propagate geometric information across any number of frames, completing missing data. For instance, for the dog in Figure 1, the chest geometry is captured only in the last few frames and is successfully completed across the entire sequence. Similarly, consistent connectivity is propagated across any number of frames. Thus for instance in the spheres sequence (Figure 4) the first five frames have the spheres glued or partially glued, but as the sub-sequences are joined together the correct topology is progressively propagated throughout. The hierarchical processing allows for straightforward parallelization of the computations at lower levels of the hierarchy.

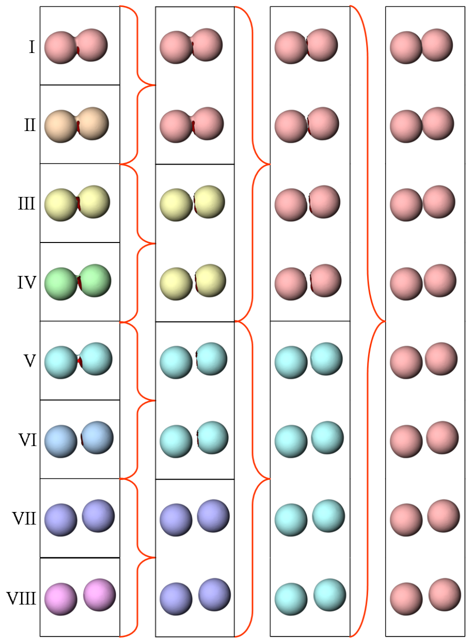

Figure 4: *Hierarchical assembly of a consistent frame sequence. From left to right are sequences of length 1, 2, 4, and 8 respectively. Note the correction step applied to frames III and IV, V and VI, and again to the first two sequences in the second column.*

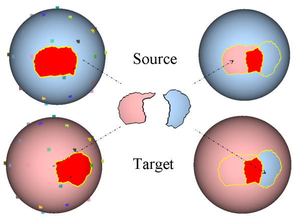

Figure 5: *Combining the source and target meshes (optical flow correspondences highlighted) to generate a consistent frame sequence. Holes highlighted in red.*

Pair-wise Sequence Combination: The main step of our method combines pairs of consecutive consistent frame subsequences into a single consistent sequence. When combining the sub-sequences we aim to obtain the union of the geometric information available in each, as illustrated in Figures 5 and 6, and to resolve any inconsistencies between them. As part of the process we also compute a common connectivity for the combined sequence, establishing an explicit one-to-one mapping throughout.

To combine the sub-sequences we first pair the last frame of the first sub-sequence, referred to as *source mesh*, and the first frame of the second sub-sequence, referred to as *target mesh*. We then propagate the combined result throughout both sequences, generating a single consistent sequence. The source and target pairing is done as follows:

- *Cross-Parameterization:* We first compute a mapping between the source and target meshes capturing the motion between them (Section 4). To this end our mapping minimizes the intrinsic change between the source and its map on the target and is consistent with the optical flow between them. Since the source and target meshes may a priori not allow for a low stretch map, we develop a specialized mapping mechanism that computes the global map as an assembly of local low-stretch maps such that areas of high stretch in the assembled mapping directly correspond to inconsistencies between the source and target (Figure 6, row two).
- *Analysis and Correction:* We use the computed mapping to locate and correct inconsistencies as follows. Consider two consecutive inconsistent meshes, such as the spheres in Figure 6. The local stretch of the mapping indicates the inconsistent regions on both but does not identify which mesh is correct and which isn't. However, if we delete the identified region on one mesh treating it as a hole and use the other mesh to complete this "missing" geometry, we will face two distinct scenarios. If we use the correct mesh to complete the incorrect one (use target to complete source in Figure 6) the mapping provided by the completion will have low stretch. In the other direction however, the map we would obtain will likely still exhibit unacceptable stretch, e.g. in the spheres example it will be effectively identical to the high stretch map in Figure 6, row two. Thus, this pairwise completion test allows us to obtain consistent reconstruction for both meshes. (Section 5).
- *Completion and Connectivity Combination:* We use the mapping to combine the geometric information from the source and target frames as illustrated in Figure 5, and generate a common connectivity for both frames. We then propagate the new connectivity and merged geometric information throughout both sequences (Section 6).

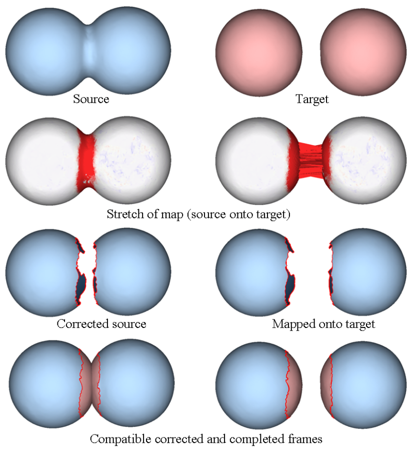
Source
Target
Stretch of map (source onto target)
Corrected source
Mapped onto target
Compatible corrected and completed frames

Figure 6: *Combining source and target frames. In row two the coloring represents stretch varying from 0.9 (blue) to 1.1 (red) (ideal stretch is one).*

Global Drift Correction: The pairwise maps we compute between consecutive frames reflect a likely motion between them. However, as we combine longer and longer sequences even tiny errors in motion reconstruction can cause some drift over time. To remove drift we apply the following pass of global correction on the final mesh sequence.

Since adjacent frames in the sequence are expected to be nearly isometric, the transformation of any given triangle from one frame to the next should be near rigid. For each triangle we compute the per-triangle transformation gradients between consecutive frames and, using polar decomposition, extract the rotational component of each gradient matrix. To improve isometry, we aim to make the transformation gradients more similar to these extracted rotations. We use the rotations as target transformation gradients for computing new positions of the vertices in the second mesh using the formulation described in [PZB*09] balancing the resulting quadratic functional with preservation of the original vertex positions. The process provides a more isometric mapping between consecutive frames, eliminating visible drift. Since all meshes have the same connectivity to obtain the new positions we only need to invert a single matrix for the entire frame sequence and back-substitute different right-hand sides, resulting in a very fast computation.

4. Cross-Parameterization

The goal of this step is to compute a map between the source and target meshes that captures the motion between them. Since we assume the motion or change to be gradual, we aim for a mapping that minimizes such change. We explicitly minimize the mapping stretch (intrinsic change) while using optical flow based initialization (Section 4.1) to bound the spatial change. The challenge we face is that the source and target meshes may not a priori support a bijective low stretch map. First, because the meshes can be incomplete, each mesh can contain regions with no corresponding counterpart on the other (Figure 5). Second, and much more challenging, the meshes may be inconsistent, with drastic intrinsic differences preventing a low stretch mapping. To overcome both problems, we compute the global parameterization as an assembly of local low-stretch maps (Section 4.2). The assembly process (Section 4.3) provides a partial mapping from source to target in which points with no corresponding map indicate regions on one mesh that correspond to missing geometry on the other, and regions of high stretch indicate local inconsistencies between the source and target.

4.1. Optical Flow Tracking

We aim to obtain a mapping consistent with the expected motion from source to target. Using the available video data, we can incorporate optical flow [Bou99] as an additional source of information on the likely motion. Unlike other methods that rely on a dense, accurate optical flow to track the geometry through time (e.g. [FP09, BHPS10]), we use sparse optical flow correspondences as soft *anchors* to initialize the parameterization. Therefore, we require only a small number of correspondences, allowing for fairly aggressive filtering of inaccurate correspondences, and can tolerate small inaccuracies in the remaining ones.

We track the optical flow between frames in all camera views, and obtain a 2D vector field describing the image-space motion of the object from one frame to the next. We then query the 3D motion of a vertex by projecting it into two cameras, advancing it through time using the 2D optical flow in camera space and projecting it back onto the model at the next frame. Next, we perform two levels of pruning to eliminate unreliable correspondences. First, we check the forward and backward flow in 2D for consistency, and only keep the optical flow samples that fall in the same pixel after moving forward and backward in time. We then check that the actual 3D point correspondences are consistent when projected back into the camera space.

For subsequent computation we aim for a set of anchors uniformly distributed across the surface, since uniform distribution allows us to use the same parameters across the entire model. To obtain such a uniform set of anchors we add extra anchors in areas where the current set is too sparse and remove redundant anchors in areas where it is too dense. We first propagate the motion vectors from the anchor vertices to all the vertices in the mesh by solving the Laplace equation using the anchor motion vectors as boundary constraints. Vertices whose motion vectors map them to the target surface are considered as secondary anchors that can be used in areas where the optical flow anchors are too sparse.

To remove redundant anchors, including most of the newly computed secondary ones, we perform a breadth-first search from all anchors and remove those within a given radius of another anchor. Whenever a secondary anchor is too close

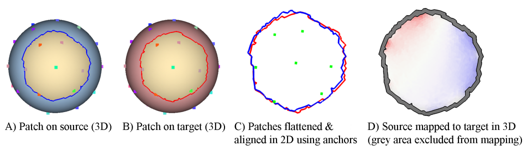
A) Patch on source (3D)
B) Patch on target (3D)
C) Patches flattened & aligned in 2D using anchors
D) Source mapped to target in 3D (grey area excluded from mapping)

Figure 7: *Local patch parameterization. The stretch range in the final map (D) is between 0.95 (blue) and 1.05 (red).*

to an original optical flow one, we delete the secondary. The number of retained anchors is on the order of three percent of the number of mesh vertices.

4.2. Local Patch-Based Parameterization

We use a local, patch-based parameterization technique as a stepping stone toward computing a global parameterization. As noted in [BPS*08], if two near-isometric surfaces are mapped to the same domain using stretch-minimizing parameterization, then their maps are likely to be identical; in other words, using the map from one to the common domain and then the inverse map from the common domain to the second surface defines a near-isometric parameterization between the surfaces (Figure 7). Based on this observation, we grow patches on both models centered around matching anchors and find patch-to-patch maps by parameterizing the patches into the plane and aligning them using the anchors. Since the patches capture very similar geometry we expect the map between them to be close to isometric and to closely resemble the actual motion between the frames in that region. This process provides a set of overlapping low-distortion local parameterizations between the two models. We then use these maps to piece together a global mapping (Section 4.3).

Patch Growth: We start by growing patches simultaneously from all the source anchors and their corresponding points on the target. To maximize patch similarity we use distance based breadth-first growth while enforcing disk-topology. The growth process terminates once the patches contain three to seven matching anchors (Figure 7, left). Three anchors are sufficient for subsequent processing, but the parameterization is more robust to noise in the optical flow when the number of anchors per patch is higher. The overall mapping stretch is likely to increase with patch size (recall that we do not expect the meshes to be truly isometric). Hence, if a patch grows to more than 4% of the total area but not enough anchors are found, the growth is aborted and the patch is discarded.

Patches grown only from source anchors may not cover the entire source model. To improve the coverage, we perform a second iteration of patch growth, growing patches from the uncovered vertices. Initially, such regular source mesh vertices do not have a known match on the target mesh. Therefore, when growing a patch from a regular vertex, we first grow the patch on the source mesh until an anchor is encountered, and then grow a patch on the target mesh around the matching anchor until its radius is equal to the radius of the first patch. From here we proceed with regular patch growth. Vertices that remain uncovered at the end of this process typically indicate source regions that lack matching target geometry.

Patch Cross-Parameterization: The two matching patches are first parameterized independently in the plane. We use ABF++ [SLMB05] for computing the planar parameterizations, because it provides a reasonable trade-off between minimizing stretch and efficiency. The two parameterizations are then aligned in the plane using an affine transformation that aligns matching anchors in a least-squares sense (Figure 7, C), implicitly providing a parameterization from one patch to the other.

For isometric patches this provides a mapping with minimal stretch. Our patches are not isometric, as the geometries are not identical and the patch growth may capture slightly mismatching regions. However, in most cases the cross-parameterization stretch (Figure 7, D) is concentrated along the patch boundaries, where the mismatch is most significant. For parameterization purposes we therefore ignore the mapping in the boundary regions (greyed out in Figure 7, D).

4.3. Parameterization Assembly

After the local parameterizations are computed, most vertices on the source mesh have multiple possible mappings on the target based on the patches they belong to. To obtain a one-to-one map, we first select for each vertex a single local mapping from this set and then apply an optimization procedure to improve the resulting parameterization.

Map Selection: Our goal is to select a patch, or local map, for each vertex, such that the resulting global map exhibits minimal stretch. This goal translates into a combinatorial optimization problem, which unfortunately is very difficult to solve. Instead we opt for the following efficient heuristic, which combined with the subsequent optimization yields the desired result. We observe that in the local mappings stretch is concentrated mostly near patch boundaries (Figure 7, D). Hence, for each vertex we select the map corresponding to the patch in which the vertex is closest to the center. This heuristic results in global parameterizations that exhibit very low stretch for triangles in the interior of most local patches, and higher stretch in the patch boundary regions. Triangles whose vertices are mapped using different patches typically exhibit the worst stretch, and in extreme cases can even be flipped (Figure 8, left).

Optimization: To improve the parameterization we need an optimization procedure that, given the initial map, simultaneously reduces the stretch and fixes flipped triangles. In the presence of flips, direct stretch optimization (e.g. [SAPH04]) often gets stuck in local minima, because sliding a vertex along the mesh in the correct target direction increases the local stretch before decreasing it again. In addition, direct stretch minimization [SAPH04] is very time consuming, requiring runtimes of an hour or more for meshes of interesting size.

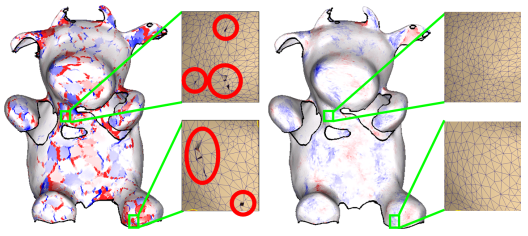

Figure 8: *Mapping optimization: (left) Initial global parameterization with stretch and flipped triangles most prominent when mapping has switched patches; (right) stretch after optimization is applied (six global iterations).*

We observe that, because the change in the intrinsic shape from one frame to the next is small, a local shape-preserving functional is consistent with stretch minimization in our setting. Rather than directly minimizing stretch, we thus optimize a functional preserving the Laplacian coordinates of the source vertices [LSA*05] when mapped to the target. Evaluating the Laplacian coordinates is significantly faster than evaluating stretch, but more importantly, the evaluated functional clearly detects flipped triangles, and minimizing it eliminates them. Since the distortion is concentrated in small regions, we can effectively utilize a local relaxation technique, iteratively moving one source mesh vertex at a time over the target mesh to reduce the shape preservation error

$$\min_{\tilde{v}} \| L(\tilde{v}) - L(v) \|_2^2 \, . \tag{1}$$

Here L is the Laplacian coordinates operator [LSA*05], v are the original positions in the source mesh, and $\tilde{v}$ the mapped positions on the target mesh. We constrain the solution space to the target mesh, thus preserving the initial target geometry. To search for a local minimum, we use a random walk approach. To speed optimization, we process the vertices in a decreasing order of their error (Equation 1). Usually only a few (five or six) global iterations are sufficient to reduce stretch to acceptable levels across much of the mesh (Figure 8, right). Once the flipped triangles are corrected, it is possible to switch to direct stretch optimization, but in our experiments this did not improve results significantly.

Areas that have local high stretch even in the optimized map are indicative of geometric inconsistencies between the processed frames (Figure 6, row two). Vertices on either model that remain unmapped at the end of this step typically indicate geometry missing in the other model and are incorporated into that model by the completion step (Section 6).

5. Analysis and Correction

The map computed in the previous step is expected to approximate the motion between the frames. Thus under our gradual change assumption, mapped triangles should exhibit low stretch and the vertex motion prescribed by the map should be sufficiently small. A violation of either of these conditions indicates a problem in the local reconstruction of

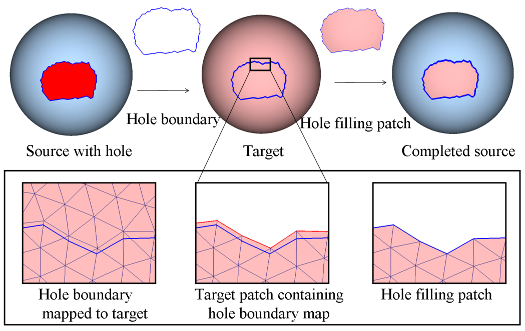

Figure 9: *Completion: (left) Mapping the boundary; (center) extracting patch (right) completing the geometry. Missing geometry highlighted in red.*

either the source or target meshes and their corresponding sub-sequences (Figure 6, row two).

Analysis: Manually setting a threshold on acceptable magnitude of vertex motion can be problematic as it strongly relates to the local motion speed. Instead, we found that this test can be robustly replaced by checking for motion similarity between neighboring vertices and between vertices and neighboring anchors. Vertices belonging to triangles that exhibit high-stretch, and vertices whose motion is much larger compared to neighboring vertices or anchors are flagged as potentially incorrect and clustered into regions.

Correction: Typically inconsistencies are caused by incorrect reconstruction of only one of the source or target sequences, but a priori we do not know which one. Deleting the flagged regions on both would satisfy the motion prior but is likely to delete valuable correct data present in one of the sequences. Therefore, we need to identify which, if any, of the source or target regions can be a viable source for geometry completion replacing the flagged incorrect data on the other. We consider a region as a viable completion source if the mapping defined by the completion process (Section 6) satisfies the gradual change criteria.

For each of source and target, we first delete the region in one mesh and complete the hole with the corresponding region from the other, using the mechanism described in the next section. We then test if the resulting mapping obeys our gradual change test. Typically the completion is viable in only one direction and the obtained completion result is used for the combined sequence. If both completions violate the gradual change test the flagged regions are deleted from both frame sequences. In the extremely rare case where both completions are viable, for simplicity, we pick the region on the source to compete the target.

6. Completion and Connectivity Combination

The last step in combining two sub-sequences into one is mutual completion, where geometric details present in one sequence but lacking in another are incorporated into the latter. of the reconstructed motion, we generate a shared connectivity for both sequences. This step simplifies further processing and allows for compact motion representation. We use the source sequence connectivity as the basis for the shared one, modifying it as necessary during correction and completion steps. We first combine the source and target meshes and then propagate the result through both sequences. To represent the target mesh using source connectivity we simply position the source vertices on the target using the previously computed parameterization.

Intuitively, completion is warranted when areas on one model correspond to holes on the other. Thus we use source regions with no mapping on the target to complete the new target mesh. Since their vertices are already part of the new target connectivity, we need only to find likely, new, target positions for them. Based on the gradual change prior, we expect the local shape of the mesh to change very little between consecutive frames, and thus the vertex Laplacian coordinates to be largely preserved. Based on this observation we position these vertices in the target frame using a standard Laplacian editing technique [LSA*05], aiming to preserve the source Laplacian coordinates of these vertices while fixing the positions of the vertices for which a mapping to the target exists. This process is very similar to the approach described in [KS05].

To identify target details missing in the source we map the source mesh boundaries to the target (Figure 9, bottom left). The regions on the target outside these boundaries represent the completion geometry. Incorporating the details in this direction is slightly more challenging since the new source connectivity differs from the original target one. To embed the identified vertices into the new connectivity we parameterize each extracted region in the plane and re-triangulate it in 2D keeping the vertex positions and connecting them to the mapped boundary edges (Figure 9, bottom right). Once the vertices are embedded into the new connectivity we proceed as before, using Laplacian editing to obtain the source mesh positions for these vertices.

Propagation: If the processed frame sequences contain more than one frame, the correction and completion need to be propagated across the sequence. The correction is straightforward. Since the sub-sequences are already compatibly meshed, we simply delete triangles removed from source or target from each of the corresponding sub-sequences. We then propagate the new combined compatible mesh across the target sub-sequence using the barycentric coordinates of the source vertices on the target. Finally we complete missing geometry across the entire sequence using the Laplacian mechanism described above. After all these steps are performed we have a single compatible mesh describing the frame sequence, explicitly defining both the geometry and motion of the captured data.

7. Results

Throughout the paper we demonstrate our reconstruction method's results on a number of datasets of varying complexity. Four of the datasets are synthetic (spheres, hand, male, and fist). These were "captured" using a virtual scanner which generates point-cloud inputs simulating occlusion and other artifacts [SAL*08]. To simulate optical flow we use known correspondences on 3% of randomly sampled mesh vertices. Using synthetic data we can fully evaluate the reconstruction accuracy by measuring the distance from the reconstruction to the original models (see Table 1). The other five datasets (t-shirt, glove, dog puppet, and two faces) are captures of real objects.

The *hand* (Figure 2) and *male* (Figure 10, top) datasets show simple motions which generate contacts, causing methods that do not account for those to fail. In both datasets, large areas of the models are occluded through much of the sequences. In the hand sequence there isn't a single frame that contains the complete geometry. Our method correctly reconstructs both, resolving the contacts and completing missing data.

The *fist* (Figure 10, row two) is perhaps our most complex example in terms of the motion magnitude and the amount of occlusions. Wand et al [WAO*09] report failure on a scan of similar motion. Our method recovers both motion and geometry, completing the missing data throughout and preserving fine features such as the lines on the palm. The example also demonstrate a limitation of the method, in that it does not explicitly prevent self-intersections and cannot predict the motion of large occluded surfaces. As a result, as the captured motion continues (see attached video) with the thumb occluded further, the reconstruction intersects the thumb and the fingers.

The *T-Shirt* sequence (Figure 10, row three) from [BPS*08] highlights the robustness of our completion mechanism and detail preservation. Even though no template is used, our results are nearly identical to those of the garment reconstruction method of [BPS*08] who use a template and other strong priors to generate the results (see attached video). The difference is only notable in areas where data is unavailable in any of the input frames and where Bradley et al. use the template for completion, e.g. under the arms.

The *glove* model (Figure 10, bottom) is reconstructed from a single binocular video pair, and shows different fingers coming into contact at different points in time. This example is typical of the type of two-dimensional self-contacts demonstrated in recent work [WJH*07,WAO*09,BHPS10]. In contrast, the *dog puppet* example (Figure 1) shows a more complex motion, comparable to that of the synthetic *fist* model, where the dog's folding paws occlude a large portion of the chest. Our method successfully recovers the paw shape as well as the occluded chest geometry in this situation. Previous methods that show results of similar topological complexity [LAGP09] required a template, while our reconstruction does not.

We reconstructed two face sequences (Figures 1, 3, 11), captured by [BHPS10], with complex changing facial expressions. The sequence in Figure 1 contains multiple self contacts as the mouth opens and closes. The specialized method of [BHPS10] fails on this example, while our general method provides satisfactory results. The sequence in Figure 11 is a good example of gradual, but substantial intrinsic change over time, as the skin on the forehead and cheeks significantly stretches between expressions. The circle pattern we use to visualize the motion reflects this stretching with the circles shrinking and stretching accordingly. In both inputs the point clouds are especially noisy in the hair region, causing some flickering in the reconstruction. [BHPS10] discard these parts of the scan manually, removing them from the first frame which they use as a template for the sequence. These two examples are quite similar in nature to the face dataset used by [WJH*07, WAO*09]. In contrast to them, our method successfully retains local facial details such as appearing and disappearing wrinkles throughout the sequences.

	approx. cloud size	approx. # anchors	# output vertices	# frames	dist to input	dist to original
Puppet	900K	1400	26814	32	0.00041	
Face (Fig. 1)	7.8M	1300	51764	226	0.00021	
Hand	800K	1000	26696	64	0.00054	0.0029
Male	800K	1100	38140	16	0.00053	0.0011
Fist	600K	1100	51384	128	0.00087	0.00091
T-shirt	1M	400	22405	64	0.00073	
Glove	1M	900	25862	256	0.00086	
Face (Fig. 11)	7.7M	1300	56646	300	0.00027	

Table 1: *Model statistics: approximate numbers of cloud points and anchors per frame, output mesh size, number of frames, and deviation from per-frame meshes and ground truth (when available).*

Quantitative Evaluation: One advantage of reconstruction from synthetic data is the availability of ground-truth in the form of the original virtually scanned models. To test the accuracy of our method we measured the deviation of our reconstruction results from these models (Table 1, last column). The numbers are average per vertex distances with models scaled to the unit bounding box. The error is less than half a percent of the diagonal. For real capture ground truth is unavailable, thus we can only quantify the impact of our processing when compared to the input per-frame meshes. The deviation of our reconstruction results from these meshes varied between 0.0002 (faces) and 0.0009 (fist) (see Table 1). The measurement ignores areas where the reconstruction corresponds to input data classified as incorrect by our analysis step. The comparison of the two metrics for the synthetic models indicates that overall our global reconstruction step introduces less error into the process than the initial per-frame reconstruction. We cannot compare our accuracy to that of previous methods, as to our knowledge none provide such metrics.

Runtimes: The reconstructed meshes have between $22K$ and $57K$ triangles. Most of the run-time is spent in the cross-parameterization and completion steps. The cross-parameterization takes between eight and twelve minutes per frame on a single Intel Xeon 3Ghz CPU. While per-frame completion is very fast (i.e. a few seconds), the completion time grows to five to ten minutes at higher levels of the hierarchy where it is propagated across longer and longer sequences. Since our algorithm is highly parallelizable, we ran it on a cluster significantly speeding up the process.

Comparison to Other Methods: The examples above include a direct comparison with [BPS*08] and [BHPS10]. However, since our method relies on the input video stream to obtain the optical flow correspondences, we cannot provide a direct comparison with methods such as [WAO*09, LAGP09] where passive video data is not available. Instead we provided a qualitative comparison, scanning comparable inputs. Our results, on complex models such as the fist, puppet and faces, appear on par with those of template-based approaches, such as [LAGP09], but without requiring a template. On inputs similar to those of [WAO*09], such as the faces, our examples demonstrate better local detail preservation. We also include examples such as the fist and puppet with more complex contact and occlusion situations than those shown by Wand et al.

8. Discussion and Conclusions

We have presented a method for globally consistent reconstruction from video based capture, using the gradual change prior. As demonstrated by the examples we robustly handle contact situations faithfully reconstructing the geometry and motion of the scanned objects. A key component of our system is a method for registering, or cross-parameterizing, meshes with similar geometry but very different topology, or connectivity. This method can potentially be used in other settings where such registration is required.

Our method has a number of limitations which can be addressed by future work.

- Since the completion propagation slows up as sequence length increases, the method as-is may become impractical for very large sequences. A potential fix would be to restrict the completion from propagating beyond a certain number of frames, trading quality for speed.
- By relying on optical flow we limit ourselves to passive-video based capture settings. Depending on the type of models the optical flow could potentially be replaced by dense feature matching.
- On very fast motion of complex surfaces the gradual change assumption may no longer hold. This problem can be ameliorated using a setup that employs a camera with higher frame rate.
- Our current implementation of geometry completion can lead to surface self-intersections when surfaces come close-by (as happens in the fist sequence). These can be detected fairly easily, but may be quite challenging to resolve.
- The main step of our method operates on pairs of consecutive frames. It might be possible to improve the reconstruction fidelity by considering a larger temporal window and additional motion priors.

Acknowledgments

We would like to thank James Gregson, Li Li and Cody Robson for their help preparing the synthetic datasets. This project was partially funded by NSERC, MITACS, and GRAND NCE.

References

[ATR*08] AHMED N., THEOBALT C., ROESSL C., THRUN S., SEIDEL H.-P.: Dense correspondence finding for parameterization-free animation reconstruction from video. In *Proc. CVPR* (2008), pp. 1–8.

[BBH08] BRADLEY D., BOUBEKEUR T., HEIDRICH W.: Accurate multi-view reconstruction using robust binocular stereo and surface meshing. In *Proc. CVPR* (2008).

[BHPS10] BRADLEY D., HEIDRICH W., POPA T., SHEFFER A.: High resolution passive facial performance capture. *ACM Trans. Graphics (Proc. SIGGRAPH) 29*, 3 (2010).

[Bou99] BOUGUET J.-Y.: *Pyramidal Implementation of the Lucas Kanade Feature Tracker: Description of the Algorithm*. Tech. rep., Intel Corporation, Microprocessor Research Labs, 1999.

[BPS*08] BRADLEY D., POPA T., SHEFFER A., HEIDRICH W., BOUBEKEUR T.: Markerless garment capture. *ACM Trans. Graphics (Proc. SIGGRAPH) 27*, 3 (2008), 99.

[dAST*08] DE AGUIAR E., STOLL C., THEOBALT C., AHMED N., SEIDEL H.-P., THRUN S.: Performance capture from sparse multi-view video. *ACM Trans. Graphics (Proc. SIGGRAPH) 27*, 3 (2008), 98.

[FP08] FURUKAWA Y., PONCE J.: Dense 3d motion capture from synchronized video streams. In *Proc. CVPR* (2008), pp. 1–8.

[FP09] FURUKAWA Y., PONCE J.: Dense 3d motion capture for human faces. In *Proc. CVPR* (2009), pp. 1–8.

[GILM07] GOLDLUCKE B., IHRKE I., LINZ C., MAGNOR M.: Weighted minimal hypersurface reconstruction. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 29*, 7 (2007), 1194–1208.

[HAWG08] HUANG Q.-X., ADAMS B., WICHE M., GUIBAS L.-J.: Non-rigid registration under isometric deformations. *Computer Graphics Forum (Proc. SGP) 27*, 5 (2008), 1449–1457.

[KS05] KRAEVOY V., SHEFFER A.: Template-based mesh completion. In *Proc. Symposium on Geometry Processing (SGP)* (2005), p. 13.

[LAGP09] LI H., ADAMS B., GUIBAS L. J., PAULY M.: Robust single-view geometry and motion reconstruction. *ACM Trans. Graphics (Proc. SIGGRAPH Asia) 28*, 5 (2009).

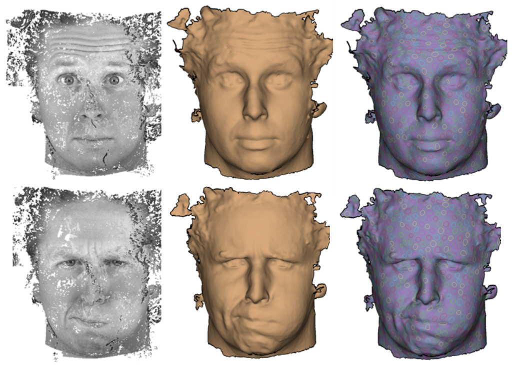

Figure 11: *Globally consistent reconstruction of another face from [BHPS10]. From left to right: textured point clouds, reconstructed geometry, circle pattern visualizing correspondences.*

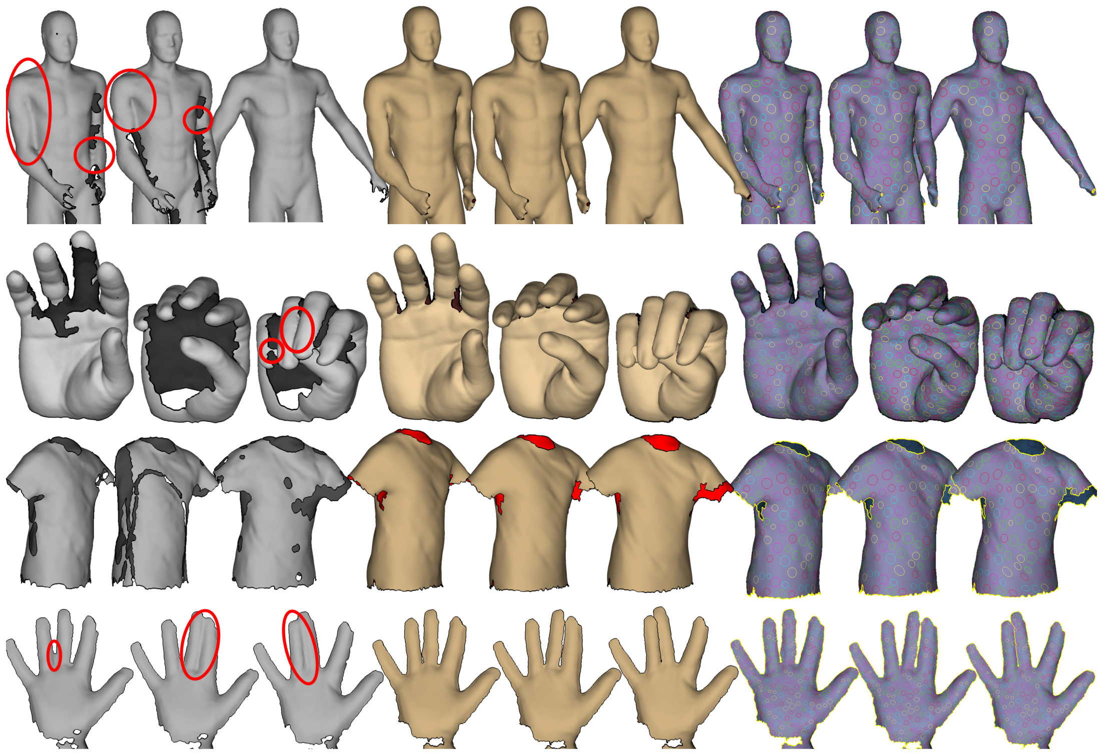

Figure 10: *Globally consistent reconstruction of male, fist, t-shirt, and glove. From left to right: individually reconstructed frames (red circles highlight inconsistencies), reconstructed geometry, circle pattern visualizing correspondences.*

[LSA*05] LIPMAN Y., SORKINE O., ALEXA M., COHEN-OR D., LEVIN D., RÖSSL C., SEIDEL H.-P.: Laplacian framework for interactive mesh editing. *International Journal of Shape Modeling (IJSM) 11*, 1 (2005), 43–61.

[MFO*07] MITRA N. J., FLORY S., OVSJANIKOV M., GELFAND N., GUIBAS L., POTTMANN H.: Dynamic geometry registration. In *Proc. Symposium on Geometry Processing (SGP)* (2007), pp. 173–182.

[PG08] PEKELNY Y., GOTSMAN C.: Articulated object reconstruction and markerless motion capture from depth video. *Computer Graphics Forum (Proc. Eurographics) 27*, 2 (2008), 399 – 408.

[PH08] PARK S. I., HODGINS J.: Data-driven modeling of skin and muscle deformation. *ACM Trans. Graphics (Proc. SIGGRAPH) 27*, 3 (2008), 96.

[PZB*09] POPA T., ZHOU Q., BRADLEY D., KRAEVOY V., FU H., SHEFFER A., HEIDRICH W.: Wrinkling captured garments using space-time data-driven deformation. *Computer Graphics Forum (Proc. Eurographics) 28*, 2 (2009), 427–435.

[SAL*08] SHARF A., ALCANTARA D. A., LEWINER T., GREIF C., SHEFFER A., AMENTA N., COHEN-OR D.: Space-time surface reconstruction using incompressible flow. *ACM Trans. Graphics (Proc. SIGGRAPH Asia) 27*, 5 (2008), 110.

[SAPH04] SCHREINER J., ASIRVATHAM A., PRAUN E., HOPPE H.: Inter-surface mapping. *ACM Trans. Graphics (Proc. SIGGRAPH) 23*, 3 (2004), 870–877.

[SLMB05] SHEFFER A., LÉVY B., MOGILNITSKY M., BOGOMYAKOV A.: ABF++: fast and robust angle based flattening. *ACM Trans. Graphics 24*, 2 (2005), 311–330.

[SWG08] SUSSMUTH J., WINTER M., GREINER G.: Reconstructing animated meshes from time-varying point clouds. *Computer Graphics Forum 27*, 5 (2008), 1469–1476.

[TBW*09] TEVS A., BOKELOH M., WAND M., SCHILLING A., SEIDEL H.-P.: Isometric registration of ambiguous and partial data. In *Proc. CVPR* (2009), pp. 1185–1192.

[VBMP08] VLASIC D., BARAN I., MATUSIK W., POPOVIĆ J.: Articulated mesh animation from multi-view silhouettes. *ACM Trans. Graphics (Proc. SIGGRAPH) 27*, 3 (2008), 97.

[VZBH08] VARANASI K., ZAHARESCU A., BOYER E., HORAUD R. P.: Temporal surface tracking using mesh evolution. In *Proc. ECCV* (2008), vol. Part II, pp. 30–43.

[WAO*09] WAND M., ADAMS B., OVSJANIKOV M., BERNER A., BOKELOH M., JENKE P., GUIBAS L., SEIDEL H.-P., SCHILLING A.: Efficient reconstruction of nonrigid shape and motion from real-time 3d scanner data. *ACM Trans. Graphics 28*, 2 (2009), 15.

[WJH*07] WAND M., JENKE P., HUANG Q., BOKELOH M., GUIBAS L., SCHILLING A.: Reconstruction of deforming geometry from time-varying point clouds. In *Proc. Symposium on Geometry Processing (SGP)* (2007), pp. 49–58.

[ZST*10] ZHENG Q., SHARF A., TAGLIASACCHI A., CHEN B., ZHANG H., SHEFFER A., COHEN-OR D.: Consensus skeleton for non-rigid space-time registration. *Computer Graphcis Forum (Special Issue of Eurographics) 29*, 2 (2010), to appear.