Exemplar-Based Hole-Filling Technique for Multiple Dynamic Objects

Matteo Pagliardini[1,2], Yasuhiro Akagi[1], Marcos Slomp[1,3], Ryo Furukawa[3], Ryusuke Sagawa[4], and Hiroshi Kawasaki[1]

[1] Kagoshima University, 1-21-24 Korimoto Kagoshima, Japan
[2] CPE Lyon, 43 Boulevard du 11 Novembre 1918 69616 Villeurbanne, France
[3] Hiroshima City University, 3-4-1 OzukaHigashi AsaMinami-Ku Hiroshima, Japan
[4] AIST, 1-1-1 Higashi Tsukuba Ibaraki, Japan

Abstract. Entire shape reconstruction of dynamic objects is an important research subject with applications on film production, virtual reality, modeling and engineering. Typically, entire shape reconstruction of real objects is achieved by combining the outcome of objects scanned from multiple directions. However, due to limitations on the number of 3D sensors enclosing the scene, occlusions inevitably occur, causing holes to appear on the reconstructed surfaces. These issues are intensified if dynamic, moving objects are considered. Volumetric and polygonal approaches exist to address these problems. Most notably, exemplar-based polygonal methods have gained momentum due to their overall improved visual quality. In this paper we propose an extension to the plain exemplar-based technique that allows for multiple dynamic objects. With our method, adequate hole-filling candidates are sampled from spatial and temporal domains and then used to synthesize likely plausible surfaces with smooth boundaries for the hole regions.

Keywords: 3D scan, hole filling, exemplar based technique.

1 Introduction

Surface reconstruction is one of the most relevant topics in computer vision, with extended applicability to computer graphics, engineering and medical imaging, to cite a few. Typically, 3D reconstruction of real scenes is achieved through either pattern-based or range-finder-based scanning techniques, both of which rely on cameras. These camera-based systems are, however, sensitive to occlusion and noise and thus prone to introducing holes on the reconstructed surface. If dynamic scenes are considered, such as a moving actor, the severity and frequency of these holes are further aggravated.

Computer vision researchers have been addressing this problem since long. Common solutions can be categorized in two groups: image-space preprocessing or surface-domain post-processing approaches. Image-based methods are more likely to mitigate the occurrence of holes due to noise, but face difficulties regarding occlusions. Surface-based techniques, on the other hand, employ exemplar-based schemes that refer to other regions of the object. Holes are then filled by

R. Klette, M. Rivera, and S. Satoh (Eds.): PSIVT 2013, LNCS 8333, pp. 242–253, 2014.

replacing them with some other region of the surface that may closely resemble the fractured region.

These surface-domain filters are more sophisticated than their image-based counterparts. One factor that contributes to their complexity is the need for extracting and evaluating features of the partially reconstructed object. In fact, it is very likely that no suitable region exists on the surface to replace a hole. All of these factors combined impose a great challenge to robust hole reconstruction algorithms.

In this paper we propose a technique that is capable of detecting potential holes in a reconstructed object and infer substitute regions for the missing parts based on an animated sequence of the object. The central idea of our method is to scan for eligible hole-filling candidate areas in different poses (frames) of the reconstructed object in case the current pose alone proves insufficient. We are, by no means, suggesting that our technique is universal and robust enough to encompass all sorts of datasets. Instead, our primary goal with this research is to establish an initial investigation on the feasibility of using temporal information contained in dynamic datasets for surface hole inference/reconstruction. We hope that more researchers will get interested in the subject and further improve on the state-of-the-art based on the foundation developed here. Meanwhile, as mentioned in the future work, we seek to continue further validating the algorithm as we manage to obtain more interesting real datasets.

2 Related Work

Range data from multiple sources can be coalesced into a single shape through polygon-based or volume-based techniques. Polygon-based efforts include synthesizing a mesh from unorganized points [1], employing deformable models encoded as a level-sets [2], and mesh tessellation/stitching from overlapped surfaces (zippering) [3]. Once the geometry of a hole is established, a better shape can be refined through a postprocessing stage.

On the other side, volume-based methods attempt to discretize unstructured points into voxels which are later used to extract a more suitable surface description. For the initial voxelization stage, signed distance fields are typically used [4–8]. Euclidean signed distance is preferable [5, 6], but the high computational cost associated to it makes it less attractive than other simplified models such as line-of-sight distance[1] [4, 7, 8]. Once voxels are obtained, the shape of the underlying object can be regularized through level-set or Graph-cut related algorithms. Finally, the resulting polygonal mesh can be extracted through one of the many variants of the marching cubes algorithm [9].

Volume-based approaches, however, are incapable of accounting for voxels that were not observed. According to the seminal space-carving method of Curless *et al.* [4], first all voxels are initialized as unseen, and those in between each viewpoint and the object are marked as empty; as the method carves the

[1] The distance between the center of a voxel and the first intersection point on the surface along the line from the viewpoint to the voxel.

volume immediately in front of the observed objects – with any unobserved voxel remaining as unseen – an excess mesh is produced along boundaries of unseen and empty volume regions. For a sufficiently large number of captured images, this excess will likely not be connected to the mesh of the target object and can thus be pruned away. However, on a more realistic scenario, untangling the excess mesh from the real mesh becomes difficult and laborious. Furukawa *et al.* refined the discrimination of unseen voxels near the object based on a Bayesian approach, producing plausible results even for very small input sets [7, 8].

Within the scope of surface-driven hole-filling algorithms, Poisson distribution and exemplar-based schemes are commonplace [10–15]. Poisson distribution is particularly useful for smooth interpolation of geometrical data along the boundaries of the hole-damaged region. Either voxel and polygon-based schemes can benefit from this interpolation: Kazhdan and Hoppe [11] rely on Poisson distribution to establish a characteristic function to assist during isosurface extraction, while Zhao et al. [15] refine the position of a linearly-interpolated points along the missing region through a Poisson distribution. These methods, however, fail to propagate high-frequency features into the hole region. Exemplar-based techniques, in contrast, can reproduce more faithfully the overall features of the object into the missing areas. Although exemplar methods are more widely adopted for 2D inpainting problems, Kawai *et al.* have shown that they can be successfully modified for 3D inpaintings as well [12–14]. A key component of Kawai *et al.* algorithm is the concept of coherency, evaluated by comparing the similarity of hole-filled shape with that of the rest of the mesh. The algorithm attempts to maximize this coherency in order to patch the holes in a more object-encompassing fashion. Compared to Poisson-based methods, exemplar-driven reconstructions tend to appear more realistic at the expense of requiring more sophisticated mechanisms for feature extraction and evaluation.

In this paper, we investigate a temporal extension to the exemplar-based technique of Kawai *et al.* [12–14]. We have refrained from investigating volume-based hole-filling approaches in the temporal domain due to the relatively scarce literature on the subject. The technique hereby described is thus completely polygon-oriented. As is usually the case, polygonal techniques tend to have lower computational costs than their volume-based counterparts.

3 Overview

The proposed method builds upon the research of Kawai *et al.* [12–14]. Missing regions are inferred through an iterative process that attempts to minimize an energy function which models the level of coherency of the inferred surface. Coherency is evaluated by comparing patches of the inferred region against other thoroughly chosen patches around the mesh. The algorithm does not promptly address strict feature recovery, but rather concentrate on propagating frequencies into the hole region. Our technique also enables partial control over the frequencies recovered by meddling with the patch size p_S.

As proposed, the technique has been extended to animated sequences of the object. This way, coherent hole filling shape can be inferred from information

contained in different time frames as well. Since the location of holes is likely to differ among distinct frames, missing regions are retrieved by looking at neighboring frames. In addition, assuming a frame rate compatible to the movement characteristics of the captured sequence, our technique can fill holes in dynamic scenes without resorting to any sort of tracking. For the extreme case where coherent shapes can not be inferred from any frame, a classic exemplar-based hole filling is performed instead.

The process begins by delimiting three regions for each individual pose of the time-varying sequence:

1. the hole region, which is initialized with a coarse shape that roughly fills the hole and will evolves at each iteration.
2. the extended region, consisting of all points for which a sphere of radius p_S extruded from them encloses at least one point of the missing region.
3. the remaining region – or data region – on which shape information is consulted during the hole filling process.

For brevity in the explanations below, we will adhere to the following notation: we note P^n as being a 3D point belonging to time frame n; $Patch(P^n)$ refers to the enclosing patch region centered at point P^n with radius p_S; we use Ω^n as the union of the points of the extended and missing regions of frame n; the superscript c denotes the index of the current frame being processed; and finally, Φ corresponds to the union of the data region of all frames currently investigated.

The rationale of the algorithm is that detailed shape information is gradually propagated from the extended region toward the hole region, while sustaining a higher degree of mesh coherency. To that end, each iteration of the proposed method performs two tasks: a matching step such that each point P_k^c of Ω^c is associated with a surface patch $Patch(\hat{P}_k^c)$, $\hat{P}_k^c \in \Phi$; and an alteration step in which points of the missing region are modified in an attempt to minimize the energy function (i.e., maximize mesh coherency).

During the first stage, a data patch $Patch(\hat{P}_k^c)$ is appointed as the best match for each hole/extended point P_k^c of Ω^c based on neighborhood similarities. This locality constraint can be elegantly modeled through the sum of squared differences (SSD) between data patches $Patch(P^m_{data\in\Phi})$ and the hole/extended region patch $Patch(P_k^c)$. A patch $Patch(P^m_{data\in\Phi})$ is a best match for a patch $Patch(P_k^c)$, i.e. $P^m_{data\in\Phi} = \hat{P}_k^c$, if $argmin_{i\in\Phi}(SSD(P_k^c, P_i^m))$ is solved for $P^m_{data\in\Phi}$. After the matching step all the shape pairs $(P_k^c, \hat{P}_k^c)$ are established.

In such terms, coherency can be thought of as the extent in which it is possible to find good matching surface candidates that fit the missing region. Mathematically, this can be expressed as a normalized weighted average of the sum of squared differences between the matching shape pairs:

$$E = \frac{\sum_{P_k^c \in \Omega^c} \omega_{P_k^c} \, SSD(P_k^c, \hat{P}_k^c)}{\sum_{P_k^c \in \Omega^c} \omega_{P_k^c}} \tag{1}$$

where $\omega_{P_k^c}$ denotes the weight associated to P_k^c. The weight of a point is determined based on its integrity: points in either data or extended region receive

a weight of 1 because their position is set and will not be further modified by the algorithm, while points in the hole region are assigned normalized weights proportional to their distance to the boundary of the hole. This acknowledges that points in the middle of the hole will be less correlated to the mesh than those close to the extended region.

In the subsequent stage, the position of the points in the hole region are modified in order to minimize the energy function. Kawai *et al.* has demonstrated that for fixed shape pairs $(P_k^c, \hat{P}_k^c)$, the energy representing their coherency can be minimized by analyzing parallel orthographic projections of all points of the missing region.

After each iteration, remeshing is performed to ascertain that the point density in the hole region remains uniform. The computational time imposed by the algorithm is directly related to the amount of SSD computations required during the matching step. Kawai *et al.* have proposed the use of curvature filtering to narrow the computations only to patches with similar curvature characteristics. In practice, however, we found that this improvement does not significantly reduce the computational time. With the assistance of the graphics hardware and some algorithmic optimizations we were able to devise a fast and efficient SSD computation routine. An overview of the algorithm is depicted in Figure 1.

4 Exemplar-Based Spatio-temporal Hole-Filling

Throughout this section we detail the proposed algorithm, including implementation-specific optimizations that allow for efficient searches of hole filling candidates over multiple frames.

4.1 Stepped Reconstruction

We have adopted the same approach as Kawai *et al.* for tessellating the initial shape of the hole region [12–14]. A coarse to fine filling scheme is employed, allowing for stepped frequency reconstruction. In short, at the very beginning, large patch sizes are used, yielding in a low density of points in the hole region; as the algorithm iterates, the patch size is gradually reduced, causing the point density in the hole region to increase. This iterative behavior implicitly induces coherent and progressive frequency propagation into the missing region. At the end of each iteration, remeshing is performed on the hole region in order to maintain the uniformity of the points for the next iteration.

4.2 Refinement Scheme for Potential Candidates

The goal of this matching step is to locate fitting surface regions that can fill the hole. More formally, this step is about finding the shape pairs $(P_k^c, \hat{P}_k^c)$ for each point $P_k^c \in \Omega^c$. In case of single frame hole-filling, the candidates are searched through the whole of the mesh. If multiple frames are considered, we can restrain the search by assuming slow movement variation compared to the

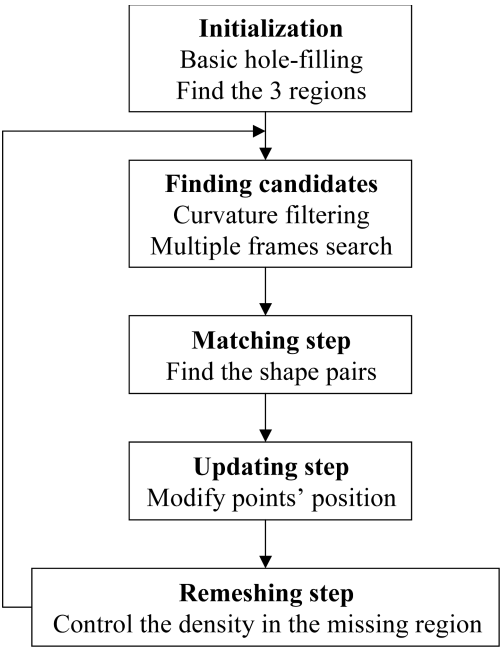

Fig. 1. Overview of the hole-filling method

overall frame-rate. For each point $P_k^c \in \Omega^c$ to be matched in the current frame c, we search within a certain number of neighboring frames. In addition, the entire mesh of each frame is not considered; instead we orient the search only to patches intersected by some sphere centered at P_k^c. We estimate the radius R of this sphere by analyzing the size of the mesh and the ratio between the rate of change in movement and the frame-rate of the sequence.

The radius of the sphere is expected to increase based on the distance of the current frame c and the queried frame m, that is, according to $|m - c|$. The time-space constraint narrows candidates from $\bigcup_{m \in TS} (D^m)$ to the restricted set $\bigcup_{m \in TS} (D^m_{|S_c^m(P_k^c)})$, where D^m corresponds to the data region of frame m, TS is the set of frames being considered, and $S_c^m(P_k^c)$ is the sphere defined previously. The set is further narrowed down during the curvature filtering step. Figure 2 summarizes the candidate refinement scheme.

4.3 Accelerating SSD Computations Using the GPU

Kawai *et al.* evaluates $SSD(P_1, P_2)$ in two steps. First, the involved patches $Patch(P_1)$ and $Patch(P_2)$ need to be overlapped. This alignment is done based on the principal directions of their respective curvatures. Once the patches

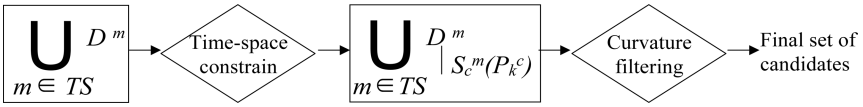

Fig. 2. Flow of the candidates refinement method

overlap, all points in $Patch(P_1)$ are orthographically projected onto the surface of $Patch(P_2)$. The SSD function is thus expressed as a normalized sum of distances:

$$SSD(P_1, P_2) = \frac{\sum_{p_k \in Patch(P_1)} \|P_k - Proj(P_k)\|^2}{N(Patch(P_1))} \tag{2}$$

where $Proj(P_k)$ corresponds to the parallel orthographic projection of P_k onto the surface of $Patch(P_2)$ along its normal direction, and $N(Patch(P_1))$ is the number of points in $Patch(P_1)$.

During each matching step, a large amount of SSD computations must be performed in order to establish the appropriate shape pairs. The computational overhead associated with patch alignment and projections is the the main performance bottleneck of the technique. In an attempt to minimize the number of SSD computations, Kawai *et al.* proposed a curvature-driven filtering step. However, we have observed that this improvement alone is not sufficient for reducing the computational time substantially.

Instead we tackle the performance issue by minimizing redundant computations. The curvature of a patch can be estimated based on its principal directions and extreme curvature values. This information can be obtained through a combination of principal component analysis and bi-quadratic fitting. Patches can then be stored and referenced directly in a curvature-oriented canonical coordinate system. This eliminates the need for the overlapping phase of the SSD since all patches naturally overlap in this space.

In addition, we propose the use of GPU rasterization capabilities for approximating ray-triangle intersections of the SSD formulation above. To that end, we render each patch (in the canonical curvature space above) to a small off-screen depth buffer. Note that we are not interested in any other pixel attribute except for depth. Conveniently, since patch rendering is subject to orthographic projection, the rasterized depth values vary linearly. As these depth buffer are rendered, we cache them in system memory and perform SSD computations on the CPU by simple image-space pixel-wise differences. The squared differences of each depth pixel are then accumulated and normalized by the size of the depth map, producing the final SSD value. The impact on performance greatly overcomes the additional memory requirements: we were able to move from about one hour in Kawai *et al.* reference implementation to just under 5 minutes.

4.4 Maximizing Mesh Coherency

After matching is performed, the energy representing the coherency of the reconstructed surface is minimized for the fixed shape-pairs found during the initial matching step. Considering all shape pairs $(P_k^c, \hat{P}_k^c)$ at once, we first translate each $Patch(\hat{P}_k^c)$ onto their respective local space in P_k^c. The overlapping is achieved using the principal directions of the curvature of $Patch(\hat{P}_k^c)$ and $Patch(P_k^c)$. Once again we use the patches stored during the curvature computation to eliminate redundant matrix calculations. When all patches overlap, all the P_k^c are orthographically projected onto these patches along their normal directions. The new position of P_k^c is obtained through the weighted average of all the resulting intersections:

$$P_k^c = \frac{\sum_{P_i^c \in Patch(P_k^c)} \omega_{P_i^c} Proj_{P_i^c}(P_k^c)}{\sum_{P_i^c \in Patch(P_k^c)} \omega_{P_i^c}} \tag{3}$$

where $Proj_{P_i^c}(P_k^c)$ is the projection of P_k^c on the surface $Patch(\hat{P}_i^c)$ translated and reoriented to overlap $Patch(P_i^c)$ on P_i^c.

Because the correlation of the missing region with the rest of the mesh increase along the iterations, we chose to modify the parameters defining the weights $\omega_{P_k^c}$ associated with the points of the missing region as the correlation changes. For the first iterations the weights of the points inside the hole region are set really small because there is no correlation between the mesh and the missing region. The weights increase gradually with each iterations, hence driving the incremental propagation of the shape inside the missing region.

5 Results and Discussions

We have evaluated our method according to three criteria: subjective visual quality, root mean square error (RMSE) and computational time. We tested our algorithm on both synthetic and real data sets. For the next results, all the meshes have been re-sized to fit a bounding cube of side 1.

5.1 Evaluation with Static Synthetic Data

To demonstrate the advantages of our temporal exemplar-based hole-filling algorithm over the pure exemplar-based method, we begin with a simple synthetic setup based upon the Armadillo model. We built a short sequence consisting of only two frames, with no apparent movement, except that the first frame contains an artificially carved hole (the second frame can thus be though of as the ground truth result). This is illustrated in Figure 3.

The result achieved by our hole reconstruction algorithm is depicted in Figure 4. There is some noticeable improvement between the standard exemplar-based method and our temporal extension. This improvement can be quantitatively confirmed based on the RMSE values in Table 1.

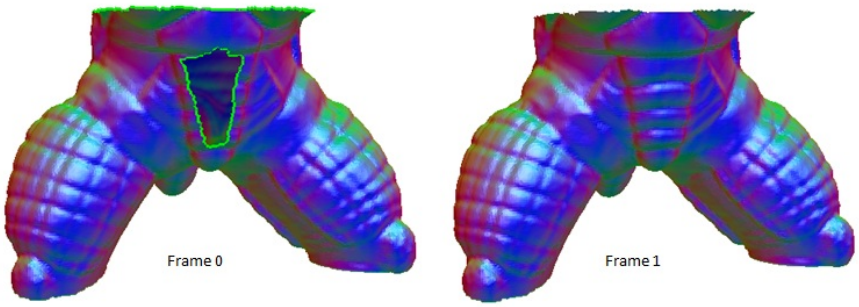

Fig. 3. Simple static data set

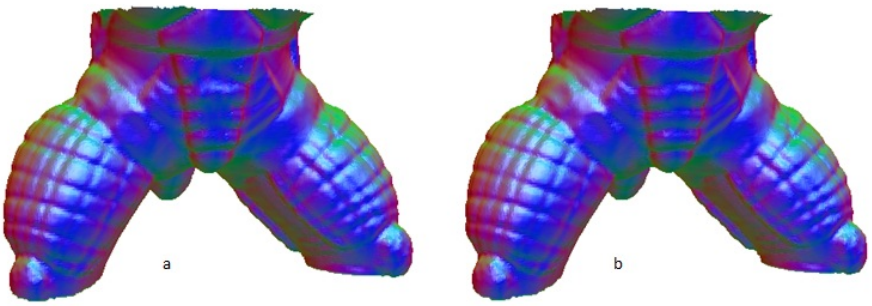

Fig. 4. a : Pure exemplar-based reconstruction i.e. only the first frame was used / b : Time-space exemplar-based reconstruction

Table 1. Evaluation of accuracy

Method	a	b
RMSE	0.00377271	0.00128592

5.2 Evaluation with Synthetic Dynamic Data

This experiment is built upon the Stanford bunny model. This time, instead of just replicating the mesh and carving a hole in one of the frames, we now alter the position and orientation of the bunny in each frame. Two kinds of hole are carved, such as at different positions over frames and the same position through the frames; meaning that there is no ground truth reference frame anymore for the latter case. Refer to Figure 5 for a depiction of this synthetic data-set.

Results of our technique on this data-set are shown in Figure 6. As can be observed, all holes were filled in a plausible, naturally consistent fashion. Table 2 lists small RSME values for this experiment, confirming the capabilities of our algorithm.

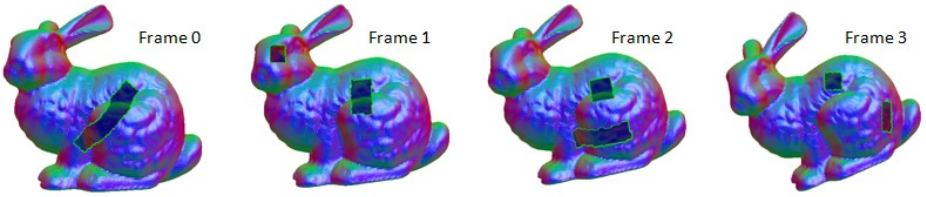

Fig. 5. Between each frames, the mesh is translated following the x axis and rotated of a small angle

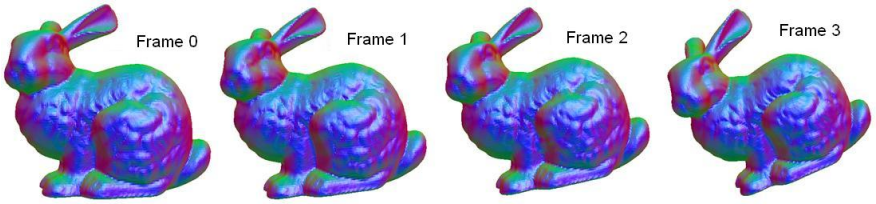

Fig. 6. Reconstructed meshes

Table 2. Evaluation of accuracy

Object:	frame 0	frame 1	frame 2	frame 3
RMSE:	0.00248812	0.00154282	0.00261936	0.000971361

5.3 Evaluation with Real Dynamic Data

Finally, we subject our algorithm to a densely reconstructed animated face. The face was captured through a scanning system built upon projected coherent light patterns. The data-set is shown in Figure 7, and results of our hole-filling technique on this data-set are available in Figure 8.

It can be observed that the majority of holes due to occlusions have been filled with appropriately coherent surfaces. Although the overall result is attractive, the technique is prone to fail if the boundaries do not give enough clues about the missing region. As a future work, we plan on driving the feature recovery using a more restrictive spatio-temporal search procedure.

5.4 Discussion on Computational Cost

The GPU-assisted evaluation of SSD substantially accelerates the algorithm. The bulk of the computational time is thus related to establishing the patches. This is further aggravated due to the fact that we rely on a dynamic, coarse to fine mesh refinement, which forces patches to be reestablished multiple times. Performance is therefore mostly constrained by the amount of density steps performed. When handling several frames, the reconstruction at each iteration is done for all frames simultaneously. This allows for significant savings on patch

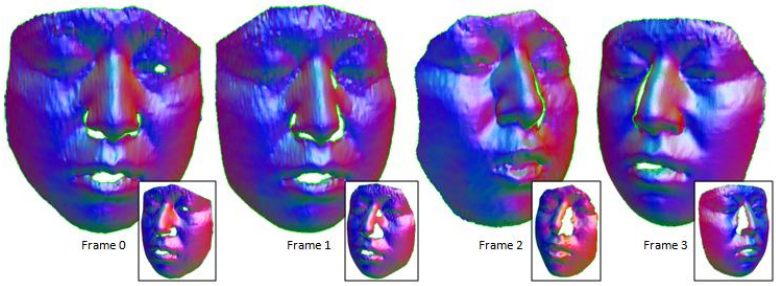

Fig. 7. The positions of the occluded parts change along the face orientation

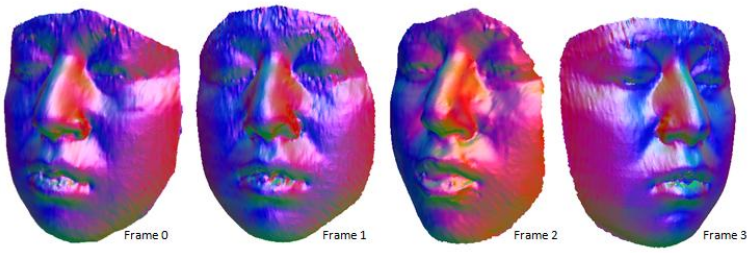

Fig. 8. Reconstructed meshes

recomputations. In other words, the overall computational time for N frames is inferior than N times the computational time of reconstructing one frame. The 4-frames reconstruction mentioned previously (figure 6) took around 7 minutes.

6 Conclusion

This paper introduced a novel approach for filling holes in dynamic scenes. The isotropic characteristic of our weighting function enables our technique to search for potential candidates in both spatial and temporal spaces. This detaches our method from object tracking, thus adding to the robustness and speed of our algorithm. In addition, even if a persistent hole exists at the same location throughout all frames, we can still fill the hole in a coherent fashion. Accuracy, performance and visual quality of the method were validated based on the experiments on static and dynamic scenes of multiple moving objects. As future research we would like to leverage the ability of the algorithm to recover features by introducing additional spatial constrains over the position of the selected patches.

References

1. Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., Stuetzle, W.: Surface reconstruction from unorganized points. In: ACM SIGGRAPH, pp. 71–78. ACM Press (1992)

2. Whitaker, R.T.: A level-set approach to 3d reconstruction from range data. IJCV 29(3), 203–231 (1998)
3. Turk, G., Levoy, M.: Zippered polygon meshes from range images. In: SIGGRAPH 1994, pp. 311–318. ACM Press (1994)
4. Curless, B., Levoy, M.: A volumetric method for building complex models from range images. Computer Graphics 30(Annual Conference Series), 303–312 (1996)
5. Masuda, T.: Registration and integration of multiple range images by matching signed distance fields for object shape modeling. CVIU 87(1-3), 51–65 (2002)
6. Sagawa, R., Nishino, K., Ikeuchi, K.: Adaptively merging large-scale range data with reflectance properties. IEEE Trans. on PAMI 27(3), 392–405 (2005)
7. Furukawa, R., Itano, T., Morisaka, A., Kawasaki, H.: Shape-merging and interpolation using class estimation for unseen voxels with a gpu-based efficient implementation. In: IEEE The 6th International Conference on 3-D Digital Imaging and Modeling, pp. 289–296 (2007)
8. Furukawa, R., Itano, T., Morisaka, A., Kawasaki, H.: Improved space carving method for merging and interpolating multiple range images using information of light sources of active stereo. In: Yagi, Y., Kang, S.B., Kweon, I.S., Zha, H. (eds.) ACCV 2007, Part II. LNCS, vol. 4844, pp. 206–216. Springer, Heidelberg (2007)
9. Lorensen, W.E., Cline, H.E.: Marching cubes: A high resolution 3d surface construction algorithm. In: SIGGRAPH 1987, pp. 163–169. ACM Press, New York (1987)
10. Bolitho, M., Kazhdan, M., Burns, R., Hoppe, H.: Parallel poisson surface reconstruction. In: Bebis, G., Boyle, R., Parvin, B., Koracin, D., Kuno, Y., Wang, J., Wang, J.-X., Wang, J., Pajarola, R., Lindstrom, P., Hinkenjann, A., Encarnação, M.L., Silva, C.T., Coming, D. (eds.) ISVC 2009, Part I. LNCS, vol. 5875, pp. 678–689. Springer, Heidelberg (2009)
11. Kazhdan, M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. In: Proceedings of the Fourth Eurographics Symposium on Geometry Processing, SGP 2006, pp. 61–70. Eurographics Association, Aire-la-Ville (2006)
12. Kawai, N., Sato, T., Yokoya, N.: Efficient surface completion using principal curvature and its evaluation. In: ICIP, pp. 521–524 (2009)
13. Kawai, N., Sato, T., Yokoya, N.: Surface completion by minimizing energy based on similarity of shape. In: ICIP, pp. 1532–1535 (2008)
14. Kawai, N., Zakhor, A., Sato, T., Yokoya, N.: Surface completion of shape and texture based on energy minimization. In: ICIP, pp. 897–900 (2011)
15. Zhao, K.H., Osher, S., Fedkiw, R.: Fast surface reconstruction using the level set method. In: First IEEE Workshop on Variational and Level Set Methods, pp. 194–202 (2001)