Interactive Modeling with Automatic Online Compression

Jean-Daniel Deschênes Philippe Lambert Patrick Hébert
Computer Vision and Systems Laboratory
hebert@gel.ulaval.ca, Laval University, Québec, Canada

Abstract

A few 3D interactive modeling systems have been developed recently. Such systems must cope with a high flow of input measurements during the entire acquisition period. Therefore, the reconstruction and rendering algorithms used must all run online. However, compression algorithms are still run offline as postprocessing. In order to develop a fully interactive modeling framework, this paper presents an online compression algorithm where the system automatically adjusts the level of detail according to the user behavior. The proposed method can reduce peak memory consumption by more than 50% during the acquisition of a typical model and the final result is comparable to offline compression. Furthermore, the results obtained show that a local acquisition approach must be prioritized.

1. Introduction

Developing an interactive modeling framework for 3D scanners has been an emerging topic in recent years [1, 2, 3, 4]. Interactive systems allow a surface model to be built and displayed in real-time while data is acquired. They are particularly well suited for hand-held sensors. Modeling is then an online interpretation process of the measurements as opposed to the accumulation of a point cloud. Producing a specific model representation, filtering, rendering or geometric compression are all modeling interpretation processes. Among these, compression is usually computed as a post processing stage [4]. However, by locally adapting the resolution to the complexity of the surface and data sampling, building a model can be made a more efficient process. For a given maximum resolution, larger volumes can be scanned. Conversely, for a given volume, a higher resolution model can be stored. The current challenge is the automation of an online compression process; that is replacing any fine resolution section by a coarser description representing the same surface within a given threshold. It is thus our aim to develop an interactive modeling framework where the resolution of the model can be adapted both *locally and automatically*. Based on the approach proposed in this paper, Figure 1 shows the state of a model being recovered during acquisition as well as the local resolution levels.

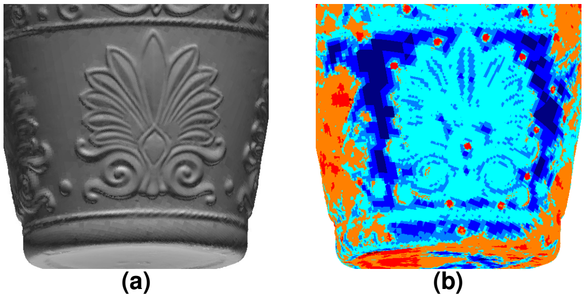

Figure 1. a) Shaded image of the model being recovered. b) Colored image depicting the resolution levels and their state. In orange and red areas, the system has detected insufficient sampling to proceed to compression. In blue areas, the system detected sufficient sampling before applying local compression. Darker blue depicts higher compression rate.

It is interesting to compare this interactive modeling process with the next best view planning paradigm [5]. In interactive modeling, a human user is part of the modeling loop and cooperates with the system. The user sees both the actual object and the model being built, i.e. the current interpretation. He also controls the acquisition strategy as well as the local sampling density on the object's surface. The modeling system must adapt to the user's behavior and provide its interpretation of the acquired data in order to support the user in view planning.

In order to adapt the resolution of the model both locally and automatically, two problems must be addressed.

First, the system must be able to automatically build a multiresolution representation from coarse to fine resolution. Local resolution on the surface must be controlled and increased depending on the measurement density. In the past, this had to be done manually [4]. Since this process must now be performed online, one must ensure sampling has reached a threshold before enabling a higher resolution level. Increasing the resolution level too rapidly would lead to inefficient memory consumption. Moreover, false apparent holes could appear in the rendered model.

The second problem to deal with is the online compression from finer to coarser levels. The system will try to compress only when local sampling is sufficient, that is when the resolution limits of the sensor are reached. Although geometric compression has been extensively studied [6], it has always been assumed that a fixed representation with all details, is initially available. In the context of online acquisition and compression, it is not guaranteed that no detail will appear before the finest resolution has been reached everywhere on the surface. Compression must then be applied locally. Nevertheless, a posteriori compression will still be useful since, by activating it, the user confirms that local model resolution is sufficient for his application. This will make it possible for the system to activate local compression where the resolution limits of the sensor have not been reached.

In the next section, we describe fundamentals for building a model representation adapted to interactive modeling. Section 3 describes the proposed approach for addressing the first problem of progressive recovery of levels of details during acquisition. Then, both online and a posteriori compressions are addressed in section 4. Simulation as well as experimental results are presented in section 5.

2. Previous work

A few systems or frameworks have been developed for interactive acquisition and modeling [1, 3, 7, 8]. Among these systems, the first general framework to produce a complete model interactively without the necessity for postprocessing was presented in [7]. In this framework, surfaces are encoded in a vector field. The vector field representation offers several advantages over other surface representations such as scalar signed distance fields or polygon meshes. Among them, the linear complexity of all modeling algorithms with respect to the number of input measurements is the most important. Nevertheless, the model is built in a single resolution vector field. More recently, a multiresolution approach was proposed where the resolution was set manually [4]. Although complete multiresolution models can be built, compression must still be processed a posteriori.

In this paper, we also build on vector fields. More precisely, a vector field is an implicit representation of a surface whose discrete version is encoded in a voxel grid. Each voxel, *V*, accumulates surface information to encode both the distance to the surface and the direction to the closest point on the surface, namely the surface normal. These values define the 3D vector ***F(V)*** at each voxel. When a 3D measurement is acquired (a point, a point with a surface tangent or a point with a surface normal), it contributes locally to the vector field recovery. To do so, each voxel encodes a covariance matrix whose eigenvectors approximate the normal and two tangent vectors at the closest point on the surface. For instance, when measurements are input as 3D points, p_i, a covariance matrix, $C(V)$, is built and updated in each nearby voxel [8]:

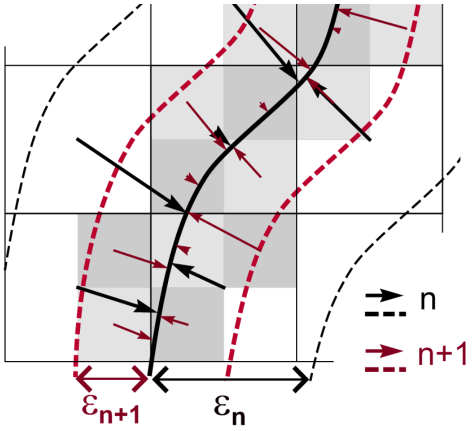

Figure 2. Multiresolution vector field within an octree structure. Two levels of detail are shown. The level n in black and the level n+1 in red. Also shown is the envelope of each level in dotted lines.

$$C(V) = \frac{1}{N}\sum_{i=1}^{N}(p_i - \tilde{p})(p_i - \tilde{p})^T = \frac{1}{N}\sum_{i=1}^{N}p_i p_i^T - \tilde{p}\tilde{p}^T \quad (1)$$

where $\tilde{p} = \frac{1}{N}\sum_{i=1}^{N}p_i$.

N is the number of measurements that have contributed to a voxel. From the right-hand side of the equation, we see that each covariance matrix can be incrementally built.

When acquired, a measurement contributes locally, within an envelope size of ε, to the recovery of a set of voxels in the field. By the same token, each voxel integrates all measurements within the envelope size. Consequently, the vector field is recovered only within this distance to the surface. The size of the envelope is set manually depending on the error level of input data. Since the covariance matrices integrate measurements, the envelope size affects the smoothness of the recovered surface. It acts as a low pass filter, a useful property for noise removal. Typically, the envelope size is set to $\varepsilon = 2$ voxels apart from each side of the surface.

The vector field framework is also particularly useful for dealing with pose refinement, namely registration, of input curves or range images [8].

One can also recover a multiresolution vector field within an octree structure [4]. The vector field is recovered at each level and the envelope size ε -in voxels- remains fixed (see Figure 2). Thus, the actual envelope size, ε, doubles from finer to coarser levels. Since the surface will be progressively discovered by locally increasing sampling density, we will recover the multiresolution field from coarser to finer levels. Each level will thus be built independently as opposed to a hierarchical reconstruction. One could think of efficiently describing a parent node - actually a voxel - matrix as a combination of its children voxel matrices but the envelope covered by the union of the child voxels at level *n+1* does not match exactly with the envelope covered by the parent voxel at level *n*. However, by building within the octree structure from coarse to fine levels with leaf nodes at different levels, it is possible to deal with a variable resolution surface. Thus, a unique multiresolution vector field makes it possible to recover a model from varying local sampling density.

The vector field representation also shows interesting properties for compression [8]. For instance, the compressed representation is not sensitive to the surface orientation with respect to the grid alignment. It can also be compressed at various levels. However, compression had to be performed a posteriori; it was still decoupled from the vector field recovery. We will explain how this process can be applied concurrently with data acquisition.

3. Automatic resolution progression

By automatically adapting the resolution level with sampling density, one avoids the necessity for considerable memory space in a scenario where all levels of the octree would be systematically recovered from coarse to fine. A given level is recovered only where sampling density is sufficient. Before introducing sampling density verification, we first describe the principle of progressive reconstruction.

For each new measurement:

1- At the coarsest level ($n = 1$), update the voxels that are closer to the measurement than the envelope size.

2- If an updated voxel at level *n* is *stable*, use the measurement to update its child voxels that are closer to the measurement than the envelope size, at level *n+1*.

3- Recursively repeat step 2 as long as the recovery of the child voxels is not in *the last active level*.

The last stage says that a measurement may contribute to the child level as long as it is *not in the last active level*. Actually, the level may reach the highest resolution level, *Nmax*, set for the sensor or a level that will have already been compressed. This latter condition says that higher resolution levels were reached and compressed afterwards where the surface is smooth. This is detailed in section 4.

Before defining a *stable voxel*, we must first define a valid voxel.

Definition: A voxel is *valid* iff the measurements within its envelope are consistent in describing a tangent plane section.

To verify that these measurements are nearly coplanar, the eigenvalues associated with the three eigenvectors of the covariance matrix are tested. The measurements are consistent if there is one smaller eigenvalue relative to the two others. The smallest eigenvalue is associated with the normal vector, the two others should be tangent to the plane. To make sure that the estimated tangent plane is valid, the three eigenvalues, e_1, e_2 and e_3, are first ordered; their sum is 1 since the covariance matrices are normalized. Then imposing the constraints $e_2 > 0.05$ and $e_1 < 0.5e_2$ validates the estimated tangent plane. A voxel might not be valid when the measurements are too noisy or when the resolution level is too coarse with respect to the surface local curvature.

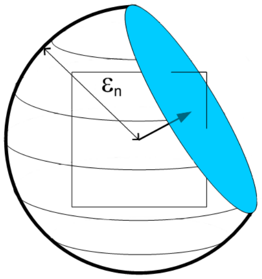

Figure 3. A voxel (square region) along with its envelope of radius ε_n. The intersection of its envelope with the tangent plane it encodes is shown in blue.

Definition: A voxel is *stable iff its sampling density is sufficient* to prevent unexpected change at its resolution level.

One must then make sure that the quantity of measurements as well as the coverage of the estimated tangent plane are sufficient within the envelope. Figure 3 depicts a voxel along with its envelope intersected by the tangent plane. In order to validate the stability, one should ideally analyze the distribution of the measurements on the estimated tangent plane. This would require storing all measurements to perform the test. Since the complexity of the algorithm must be kept low, the coverage must be approximated. Conversely, one could simply test the number *N* of measurements that have contributed to the voxel. This supposes that the intersection area of the plane with the envelope is of constant size.

A better condition is to assess the density of data based on *N*, the number of measurements acquired, as well as on the area of the intersection of the estimated plane with the envelope sphere. The value *N* is thus divided by that area before thresholding. A voxel whose estimated tangent plane is farther from its center will thus require less measurements to stabilize. Nevertheless, for *non valid* voxels, the tangent plane does not describe the actual surface well; in this case, it is imposed to accumulate the maximum number of measurements as if the tangent plane was coincident with the voxel center. Using this condition, a valid voxel may stabilize more rapidly.

The same threshold is applied at each level. This imposes a sampling density that is four times higher at each resolution level. This is well suited for a sensor that outputs uniformly sampled points. Nevertheless, setting the threshold is still empirical. It depends on the noise level, the acquisition rate and the type of sensor. When sampling is not uniform along all directions on the object's surface, the threshold should be adapted. For instance, a laser range scanner projecting stripes with a high linear density of measurements may produce a lower density along the axis of motion. This will depend on the velocity of the sensor relative to the surface. Surface sampling density varies according to the density of profiles. One can then adjust the threshold at each level considering a uniform density at the finest level. The threshold would be doubled at each level starting from the finest level. It is then assumed that the density along the profile is preserved and only the number of profiles is reduced. A severe threshold will slow down the progression. Typically, the density threshold is set between 4 and 10 points per voxel area unit when $\varepsilon = 2$.

Identifying stable voxels is useful to better control memory and make it possible to vary the density of measurement on the object's surface while displaying a consistent surface without holes due to undersampling. To display the state of a voxel at a given level, its parent voxel must have been stabilized beforehand. Moreover, the voxel must be valid and stable before rendering. This will cause an actual hole in an object to appear only when its size exceeds the size of the envelope at a given level.

During the course of progressive reconstruction, it is important to note that a newly activated voxel is initialized with $N = 0$ measurement. Since measurements used to recover coarser levels are not stored, they do not participate in the recovery of their child nodes. Moreover, using the coarser voxel covariance matrix to initialize the matrices at level $n+1$ might introduce a bias in the representation since their envelope size differs. One must therefore acquire an additional number of measurements. To assess this overhead, assume that the surface area of the object is m voxel units and that a minimum of K measurements per voxel are required. Then the minimum number of measurements that must be acquired at the finest level is mK. At the parent level, the surface area is m/4 voxels since the size of the voxels doubles. At the next level, the surface area will be m/16 voxels. Further assuming the same progression and the same minimum number of K measurements at all levels, leads to approximately 33% more points when the whole surface must be sampled at all levels. Note that in the context of interactive modeling, more data can be acquired at low cost. For instance, the hand-held device we use provides approximately 20 000 points per second.

Finally, in order to set the highest resolution level, *Nmax,* for the sensor, one will consider the error level produced by the acquisition system including registration error. In practice, *Nmax* is set such that it corresponds to a minimum voxel size of 2 to 3 standard deviations in the measurement. With a minimum envelope size of 2 voxels, this corresponds to $\pm 6\sigma$ at the highest resolution level. Progressive activation of child voxels makes it possible for the system to adapt the reconstruction level to the user behavior. Once the *Nmax* level is reached, compression is locally activated.

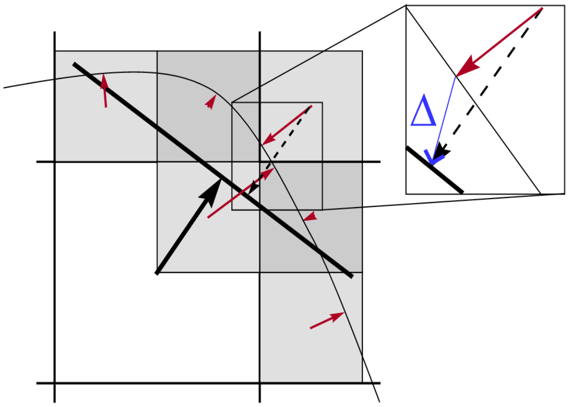

Figure 4. An object surface section encoded at two different levels of detail. The vector Δ encodes the discrepancy between both representations of the surface.

4. Vector field compression

Figure 4 shows that the same surface section can be encoded by a voxel at level n and redundantly by a set of voxels at level $n+1$. While the former voxel at level n is called the coarse voxel of that surface section, the set of voxels at level $n+1$ are said to be the finer voxels. If both the coarse voxel and the finer voxels describe the same tangent plane within a given tolerance, that is if they meet a given fitting condition, this duplicate data can be compressed. After compression, we will refer to the coarse voxel as a *compressed voxel*.

The fitting condition makes it possible to compare the surface descriptions encoded at the two levels $n+1$ and n. To do so, let the vector $\boldsymbol{F}^{n+1}(V)$ be the vector between the center of the finer voxel and the closest point on the surface at level $n+1$. Let also $\boldsymbol{F}^{n}(V)$ be a vector originating from the same voxel center but reaching the closest point on the surface at level n. As shown in Figure 4, the norm of the difference between the two vectors encodes the discrepancy between both representations. Before compressing a voxel, one must consider all finer voxels that overlap with the tangent plane section defined by the coarse voxel.

Formally, the covariance matrix C encoded in the coarse voxel at level n approximates locally an object surface section by a tangent plane $\mathcal{A}$ of circular shape. The radius of $\mathcal{A}$ is set equal to the voxel diagonal. As shown in Figure 5, the *surface overlap region* (SOR) of a voxel at level n is a cylindrical region of cross-section $\mathcal{A}$ and length $2\varepsilon_{n+1}$ centered on ***F(V)***. Every voxel whose center lies within the SOR will be tested before compression. One should keep in mind that these voxels are not necessarily all children of the coarse voxel in the octree structure.

The next two sections adapt these principles to online and offline compressions.

4.1 Online compression

An online compression scheme must deal with measurements being added progressively to the model. This makes it difficult to determine when the system should start compressing the model. Starting with the compression too early would produce a model that misses some of the actual details of the object while starting too late would result in memory waste. In order to solve this issue, when a new point is added to the model, the following four conditions must be met before compressing a coarse voxel.

1- The coarse voxel at level *n* is *valid* and *stable*.

2- All voxels in the surface overlap region at level *n+1* are *valid* and *stable*.

3- All voxels in the surface overlap region at level *n+1* are of the *last active level*, i.e. either of the *Nmax* level or having compressed children.

4- All voxels in the surface overlap region respect the fitting condition.

The first condition implies that the coarse voxel describes a tangent plane section for which we have a good confidence level (dense and consistent). Therefore, the compressed model should correctly describe the surface.

The second condition ensures that the local approximation $\mathcal{A}$ at level *n* of the object surface section is entirely covered by valid and stable voxels at level *n+1*. First, this ensures that the fitting condition is relevant everywhere on the surface section $\mathcal{A}$. Secondly, when this condition is met, the addition of further points to the model should not significantly modify the result of the fitting condition. At this stage, a finer voxel already removed by the compression process is considered valid and stable. Also, this voxel will be ignored for the remainder of the compression process.

The third condition has the following implications. First, no compression will begin before the density of points is high enough to recover a stable surface at the *Nmax* level. This ensures that a decision is made only when the surface representation has locally stabilized at the finest level of resolution *Nmax*. Failing to do so could prevent the modeling process from acquiring details that are visible only at the finest level. Secondly, after the compression process has started, a voxel at a level coarser than *Nmax* can be involved in the compression process if its children have been already compressed. Such a voxel is now a leaf node of the octree and considered of the last active level.

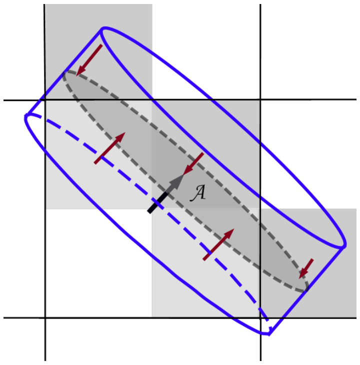

Figure 5. Illustration of the surface overlap region (in blue) for a voxel at level n.

The fourth condition ensures that the compressed model still encodes the same object surface after compression within the tolerance of the fitting condition. This tolerance is a percentage, typically 10%, of the voxel size. Finally, since the four conditions are tested each time a new point is added, the compression starts as early as possible.

Once all four conditions are met, the coarse voxel can be compressed. Its child voxels in the octree are not needed anymore and are thus removed. The coarse voxel is then tagged as being compressed as well as being from the last active level; it already contains the surface description that was directly built from measurements.

4.2 Offline compression

The online compression scheme must deal with points added progressively to the model. However, once the user is satisfied with the model, it is possible to use an additionnal offline compression scheme that does not face this constraint. It is possible to compress areas that have not reached the *Nmax* level. This is possible since no more points will be added to the model. To apply offline compression, leaf nodes that are not valid or stable are first recursively discarded. They do not represent a meaningful surface and they no longer have the possibility of changing their status.

One can now compress a given level even if its SOR is not totally covered by valid and stable voxels. In that sense, condition 2 is relaxed. We can then test conditions 3 and 4 on the valid and stable voxels that are included within the SOR. At this stage, a voxel without children is considered a voxel of the last active level.

5. Experiments

The previous concepts are validated by both simulated and experimental data sets. The first data set is generated from a high resolution model of a mannequin head. Using a profile range sensor, acquisition was simulated in order to illustrate the proposed modeling steps.

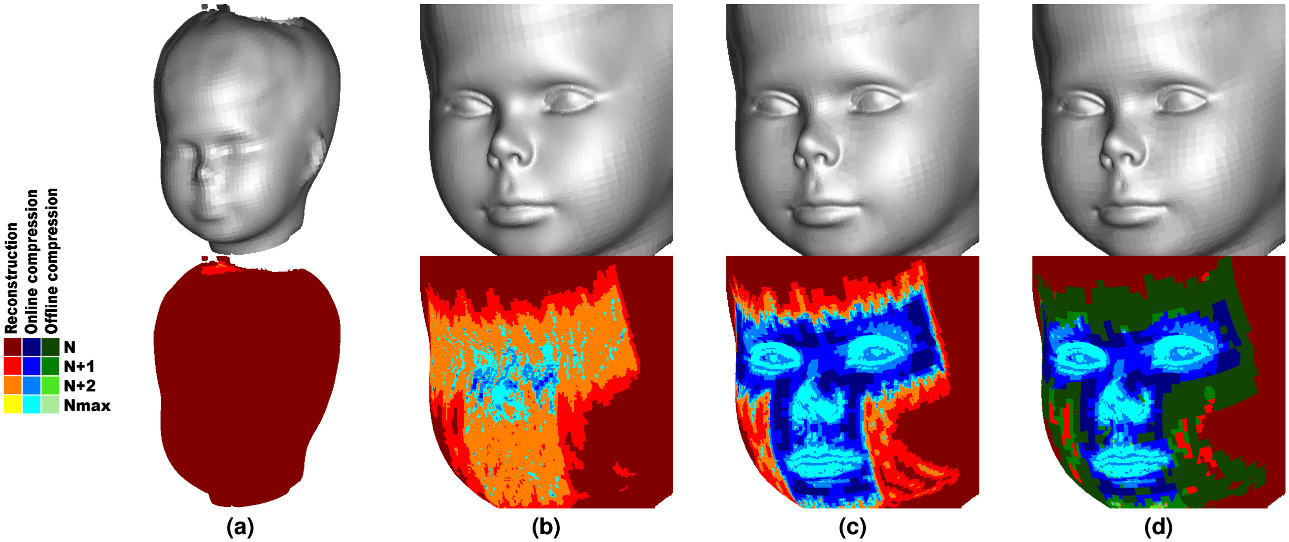

Figure 6. Model displayed to the user in a typical acquisition session. The top rows shows the model in shades of gray. The bottom row shows the same model according to the local resolution level. a) Context acquisition. b) Acquisition of a detailed region. c) Model state at the end of acquisition. d) Final offline compressed model. Tints of red are used for the reconstruction process, tints of blue are used for online compression and tints of green are used for offline compression. Darker shades are associated with coarser levels of detail.

Figure 6 presents the output displayed to the user during the session. The top row shows a shaded version of the model displayed at different time intervals. The images are rendered by applying a ray casting technique of the vector field [4]. The bottom row is an alternative display that shows the same model according to the local resolution level. On the one hand, red color tints correspond to surface areas that have not reached the *Nmax* level yet. The darker the red, the coarser is the level. On the other hand, blue color tints appear when the compression process has started. Darker blue areas are further compressed.

As shown in Figure 6a the first step consists in the context acquisition. The bottom of Figure 6a shows that the model is being recovered at a coarse (dark red) level because of the low density of points available at that stage. One can note the expected low level of detail around the eyes, nose and mouth.

Based on these visual feedbacks, the user can then plan data acquisition in order to obtain a model that encompasses details only where desired. Therefore, as a human user would do, the simulation increases the density of points around the features of interest. This is seen where finer levels (light red tones) appear in figure 6b. As expected, the system also adapts by starting the online compression of the areas that have reached the finest level as shown in blue.

Figure 6c shows a satisfactory model once the data acquisition is completed. Smooth areas around details were compressed online at a coarser level of resolution (dark blue). One can note that the actual fine details of the object surface are preserved by the compression process since they still appear in light blue. The only remaining step is the optional offline compression. The corresponding results are shown in figure 6d. Smooth regions that did not reach the *Nmax* level during acquisition, are now correctly encoded by coarse levels and appear in green.

The previous simulation showed the proposed acquisition stages within the interactive modeling framework. Its main particularity is that online compression is applied locally and progressively. We now compare three types of acquisition techniques: uniform sampling without compression, uniform sampling and finally piecewise sampling both with online compression. In order to do so, we use data from the Stanford Bunny model.

The blue curve in Figure 7 shows the progressive memory requirement in voxels according to the number of input points for uniform sampling without compression. In this case, 3D points are added to the model progressively and they are uniformly sampled over the entire surface. Since there is no compression, the memory requirement is continually increasing except when the model is completely recovered at its finest level. One can note that the memory requirement converges toward a maximum, a particularity of a representation encoded in a grid compared to point cloud. This curve, which represents the worst case, will thus serve as a reference for the other acquisition schemes.

The red curve in Figure 7 shows the same relation but for the uniform sampling acquisition scheme with online compression. One can observe that the memory requirement for such a procedure reaches a maximum as high as the previously described scheme before compressing. This is clearly suboptimal since it still requires having the complete model at its finest level in memory as did the scheme without compression.

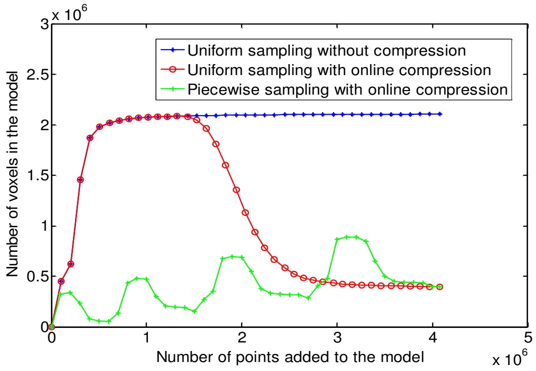

Figure 7. Memory requirements according to the number of acquired measurements while reconstructing the Bunny model for three different acquisition schemes.

This observation motivates the acquisition scheme that was presented for the mannequin head. A significant improvement in memory consumption is obtained by adding local sections one after the other. In this case, each section reaches the finest level thus allowing it to be compressed independently. The green curve in Figure 7 shows the corresponding gain in memory. It was obtained by adding 3D points to the model from four different sections uniformly while progressively sampling them, thus explaining the four visible peaks in the curve. This shows that a local approach must be prioritized to ensure memory optimisation during object acquisition.

Table 1 further motivates the proposed acquisition framework. The first column shows the voxel count for each model at the end of the acquisition, before the offline compression stage. Both uniform and piecewise sampling provide a significant compression rate of more than 80%, as displayed in parentheses.

The second column compares the maximum amount of memory used at any time during the acquisition for each technique. We can observe that a local acquisition approach can significantly reduce the maximum memory requirement needed to generate a model compared with uniform sampling. Specifically, the piecewise approach reduces the peak memory consumption by 57% over a uniform sampling.

	final voxel count	max voxel count	offline compression
Without compression	2.11M	2.11M	0.343M (83.7%)
Uniform sampling	0.396M (81.2%)	2.08M (1.1%)	0.317 (84.9%)
Piecewise sampling	0.394M (81.3%)	0.88M (57.9%)	0.316 (85.0%)

Table 1: Memory requirements for three acquisition schemes. The figures are the final voxel count for each model, the absolute maximum voxel count and the voxel count after offline compression.

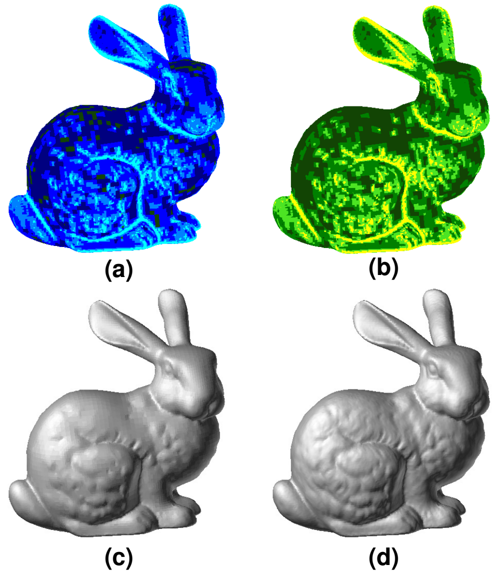

Figure 8. a) Local resolution level of the Bunny model after offline compression obtained from the piecewise acquisition scheme b) without online compression. c) Shaded model based on the representation in a). d) Reference model (without compression).

The third column shows the compression rate after applying the offline compression to each model. The additional increase in compression rate is around 3.5% for both sampling techniques compared with using only online compression. Figures 8a and 8b show the model after offline compression for the piecewise approach and the model generated without compression respectively. Tints of green represent areas compressed by the offline procedure. Figure 8c shows piecewise offline compressed models and Figure 8d shows the uncompressed model.

It is interesting to note that the model generated after offline copression and without online compression does not reach the same compression rate as those using a prior online compression scheme.

This difference shows that it is not always possible to determine exactly if a given voxel is stable during online compression. We can consider the reconstruction of the model without online compression as being more strict in applying the stability condition; all available points contribute in assessing voxel stability. Therefore, before starting compression, some voxels might have been considered as details that the online compression would have missed. Those additional details encoded in the field explain why the compression rates slightly differ.

Figure 9 shows similar results obtained with the piecewise approach on real data obtained using the Creaform's HandyScan. The sensor provides 3D points in a global coordinate system. The sensor error thus includes registration error. The data is intrinsically filtered by the vector field. The level of filtering is set by the envelope size which was 2 voxels and the minimum voxel size at level *Nmax* was 1.3 mm. The height of the vase is approximately 700 mm.

The proposed acquisition session consists of two steps: the context acquisition followed by the capture of local details. Figure 9a shows the context acquisition of the model. During this stage, one can still observe that the density of points is lower than that required to start the online compression. Figure 1 shows the state of the model after the addition of a first detail. The high density of points in this area now allows the online compression to be activated. Smooth areas around the details are thus compressed and appear in dark blue. This again shows the benefits of the proposed approach. In the next step of the acquisition process, a new detail is being added to the model as shown in Figure 9b. Once again, we observe new smooth areas that are now compressed.

The tested implementation is not optimized for memory access. More specifically, access to voxels in the surface overlap regions is particularly demanding and will gain to be optimized by adapting the octree structure. Currently, more than 2000 points per second can be processed on a desktop computer. Nevertheless, the complexity of the algorithms remains constant or, equivalently, linear with time; this is a fundamental constraint for an interactive framework.

6. Conclusion

We have contributed towards completing the interactive modeling framework with an automatic level of detail detection as well as online compression. The density of measurements gathered by the user in an area serves to determine the level of detail used to represent the object surface. This is coherent with the whole idea of interactive modeling, i.e. giving an active role to the user in the modeling loop. The user can optimally manage the available time and memory. It was possible to reduce the peak memory consumption by as much as 57% during the acquisition of a typical model. Moreover, the obtained results show how prioritizing a local acquisition approach further optimizes the available resources. Currently, the finest level *Nmax* is still manually set by the user and the SOR analysis can be further optimized.

Acknowledgments

This work has been supported by NSERC (Natural Sciences and Engineering Research Council of Canada). The authors are grateful to Annette Schwerdtfeger for proof reading.

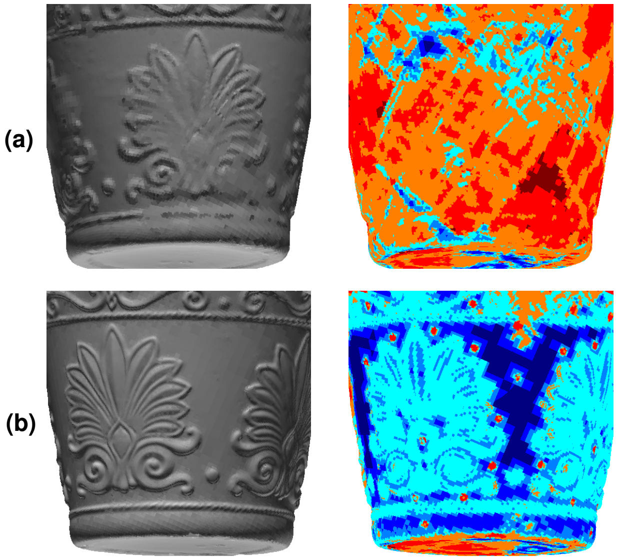

Figure 9. Two stages during modeling of a vase : a) context acquisition. b) Acquisition of two local details. The left colum displays the model in shades of gray.

References

[1] S. Rusinkiewicz, O. Hall-Holt and M. Levoy, Real-Time 3D Model Acquisition, *ACM Transactions on Graphics*, vol. 21 no.3, July 2002, pp. 438-446.

[2] F. Blais, M. Picard and G. Godin, Accurate 3D Acquisition of Freely Moving Objects, *in Proc. of the 2nd International Symposium on 3D Data Processing, Visualization and Transmission*, Thessaloniki, Greece, Sept. 2004, pp. 422-429.

[3] T. P. Koninckx, *Adaptive Structured Light*, PhD Thesis, Katholieke Universiteit Leuven, Belgium, May 2005.

[4] J.-D. Deschênes, P. Hébert, P. Lambert, J.-N. Ouellet and D. Tubic, Multiresolution Interactive Modeling with Efficient Visualization, *in Proc. of* the *Fifth International Conference on 3-D Digital Imaging and Modeling*, 2005, Ottawa, Canada, June 2005, pp. 39-46.

[5] W. R. Scott, G. Roth and J. F. Rivest, View Planning with a Registration Constraint, *in Proc. of the Third International Conference on 3-D Digital Imaging and Modeling*, Quebec, Canada, May 2001, pp. 127-134.

[6] A. Khodakovsky, P. Schröder and W. Sweldens, Progressive Geometry Compression. *in Proc. of ACM SIGGRAPH 2000,* New Orleans, USA, Jul. 2000, pp. 271-278.

[7] D. Tubic, P. Hébert and D. Laurendeau, A Volumetric Approach for Interactive 3D Modeling, *Computer Vision and Image Understanding,* vol. 92, 2003, pp.56-77.

[8] D. Tubic, P. Hébert, J.-D. Deschênes and D. Laurendeau, A Unified Representation for Interactive 3D Modeling, *in Proc. of the 2nd International Symposium on 3D Data Processing, Visualization and Transmission*, Thessaloniki, Greece, Sept. 2004, pp. 175-182.