Multiresolution Interactive Modeling with Efficient Visualization

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

Abstract

3D interactive modeling from range data aims at simultaneously producing and visualizing the surface model of an object while data is collected. The current research challenge is producing the final result in real-time. Using a recently proposed framework, a surface model is built in a volumetric structure encoding a vector field in the neighborhood of the object surface. In this paper, it is shown that the framework allows one to locally control the model resolution during acquisition. Using ray tracing, efficient visualization approaches of the multiresolution vector field are described and compared. More precisely, it is shown that volume traversal can be optimized while preventing holes and reducing aliasing in the rendered image.

1. Introduction

After more than 20 years of developments in 3D range sensors and 15 years of significant advances in modeling objects, the current challenge is producing the final result in real-time while data is collected. This new field of *3D interactive modeling* integrates the processes of scanning the surface, incremental surface reconstruction, alignment, fusion, as well as compression and visualization. The whole process is performed in a feedback loop where a user or a mechanism typically selects the sensor viewpoint and positions the sensor based on the current state of the surface model.

This paper deals with two specific objectives in 3D interactive modeling:

1) **Providing 3D interactive modeling with varying resolution of surface reconstruction.** By locally controlling the level of details (resolution), interesting features can be reconstructed with higher resolution than surrounding context. This allows faithful modeling without unnecessary storage space and scanning time overhead. To achieve such reconstruction, the modeling process must cope with varying sampling density. Features are densely sampled and reconstructed with high resolution while the surrounding context is sparsely but rapidly sampled.

2) **Improving the quality and speed of online model visualization.** So far the main goal in visualization was to provide visual feedback in order to ensure full coverage of the object's surface. By improving the quality of visualization during interactive modeling, it becomes possible to assess the quality of the reconstructed model as well.

Surface representation is the corner stone in interactive modeling. Among the various representations including point clouds, polygons and volumetric cells, only the latter appears adequate in the most general situation. However, it is worth mentioning that in recent work [1], a combination of these representations was proposed where lists of points are accumulated in volumetric neighborhoods in order to build a triangulation in real-time. This framework is interesting although limited for some steps such as alignment and its capability to locally accommodate variable surface sampling. In the last decade, volumetric representation has evolved progressively to encompass the whole modeling chain. Hoppe *et al.* [5] addressed the surface reconstruction step from a point cloud using a volumetric grid. An implicit surface function is generated and input to a marching cubes algorithm to produce a mesh that is rendered. Hilton and Illingworth [4] as well as Curless and Levoy [2] pushed the framework further in the field of 3D imaging from range data. Although the interest was not real-time modeling, it was possible to easily merge data in the form of range images and reconstruct a surface. Improving the representation with surface normal estimates was essential to allow online approximate range image alignment (registration) and visualization by Rusinkiewicz *et al.* [9]. Surface reconstruction is completed off-line before visualizing the final result.

Tubic *et al.* [11] have recently proposed a volumetric framework where all modeling steps are integrated and their complexity remains linear with respect to the number of data. The framework also allows the integration of any type of range data including points, curves and surface sections [12] while benefiting from the structure, namely tangents, of the input data when it is available. The surface is represented as a vector field encoding the surface normal and the closest point at each voxel in the neighborhood of the surface.

In this paper, this framework is extended to better exploit multiresolution reconstruction for a quick capture of the context. It is shown how the multiresolution structure may adapt locally to allow integration and fusion of measurements. Intermediate levels are preserved to ensure continuity in resolution. The main challenge is to visualize the surface representation state with high quality as fusion is performed. To do so, three different visualization strategies are analyzed; these are variants of ray tracing in a volume. Avoiding holes due to multiresolution discretization as well as aliasing for visible surface contours are the main concerns. A foveal strategy for improving visualization speed and progressive display is also described.

The principles of surface modeling in a vector field are first recalled before the multiresolution issue is discussed in section 3. From the computer graphics literature, related work on surface visualization is addressed in section 4 along with the rendering strategies adapted to interactive modeling. Although intermediate results are presented throughout the paper, additional results follow in section 5.

2. Why a vector field surface representation?

If only one step in the modeling process is not of linear computational complexity with respect to the number of measured points, the system slows down as the amount of range data increases and eventually, becomes too slow to be usable. Most of the computational complexity issues have proximity operations, closest point search for example, as a common denominator. This is not only the case for registration which is often based on the Iterative Closest Points (ICP) algorithm but for surface reconstruction as well: surface reconstruction can be seen as the operator that computes points on the final model from the nearest measured points. It was shown earlier [11][12] that reaching the linear complexity depends not only on algorithms but on adequate surface representation as well.

A particularly well suited surface representation that respects computational complexity constraint, is vector field surface representation. The vector field is an implicit representation defined on a regular 3D lattice of points that encodes *direction* and *distance* towards the closest surface point. This is equivalent to encoding the tangent plane at the closest surface point (see Figure 1). Vector fields allow surface reconstruction from range images, range curves, unorganized sets of points or any combination of these. Furthermore, vector fields allow efficient registration, visualization and compression, all of them with linear computational complexity. The only exception is visualization that should have a complexity independent of the model size since a snapshot computed from the whole model must be rendered at any time. When the model is completed, a triangulation can be produced if necessary.

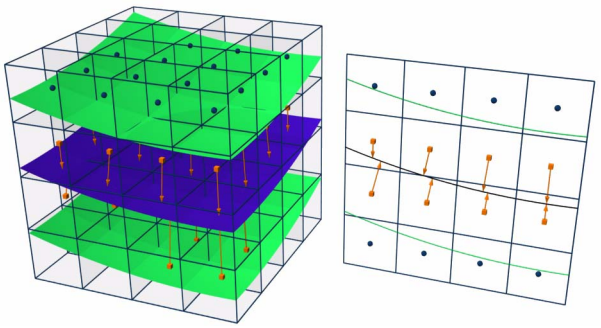

Figure 1. Example of the vector field. a) The vector field is computed on a regular grid of points (voxel centers) so that the vector at each voxel center points towards the closest surface point. The field is defined only in the vicinity of the surface, that is in the volumetric envelope (region bounded by two iso-distant surfaces, colored in green). b) A cross section of the vector field.

2.1 How does it work?

In the vector field framework, surface reconstruction means reconstructing a vector field representation from the range data. Regardless of the type of range data, vector field reconstruction always proceeds in the same way. The tangent planes encoded by vector fields are incrementally reconstructed using either relative positions of points, local surface information contained in normals (range images), or tangents (surface curves). Exploiting local surface orientation contained in the range data, surface tangents or normals makes the surface reconstruction and registration less sensitive to sensor positioning errors [12].

Theoretically, the vector field can be obtained as the derivative of the scalar field. Nevertheless, numerical differentiation can be avoided by directly constructing the vector field from input data. Incremental reconstruction is thus achieved by updating a covariance matrix from tangents, normals and relative positions of points at all voxels in an envelope within a predefined distance around the measured range data. Linear complexity with respect to the number of measured points is achieved by defining a local volumetric neighborhood, namely the fundamental cell [11] that selects voxels to be updated. The fundamental cell is different for each element of the input data: a point, line segment or a triangle. Since such a cell contains only voxels affected by corresponding data elements and since its shape can be computed independently for each element, the computational complexity is linear with respect to the amount of input range data.

Matching closest points, the most computationally expensive part of registration, is solved trivially using vec-

tor fields. Since the field is computed on a regular grid it is easy to find the closest voxel for any point. The closest voxel on the other hand directly encodes the closest tangent plane. The corresponding point is the closest point on the tangent plane.

When necessary, compressing the vector fields is also conceptually very simple. Since vector fields encode tangent planes, they are actually constant in planar regions of the surface. Unlike scalar distance fields that are never constant due to the change of the distance to the surface, vector fields can be compressed simply by locally replacing groups of voxels that encode the same tangent plane with a smaller number of larger voxels.

When it comes to the visualization, the complexity problem is related to the ever increasing number of points during the acquisition of data. If all acquired data is displayed using a renderer, the complexity increases as the number of points increases, which slows down the visualization process. It is therefore essential to use a model that merges and/or discards points in order to limit the size of the data. Being defined on a finite and regular grid, vector fields allow direct ray tracing with worst-case constant complexity when the smallest voxel size is set. The principle is simple, the surface is viewed where the ray intersects with the tangent encoded in voxels. There is no absolute need for an intermediate (polygonal) representation.

Using the vector field representation, one defines the whole volume size to include the object as well as the maximum resolution level (minimum voxel size). It is then guaranteed that the whole modeling loop will not degrade after a long (infinite) acquisition time. For each step, the complexity is constant (linear with time or the number of input data) and the constant depends on the envelope size. The size of the envelope is typically 1 or 2 voxels around surface data. It depends on related factors such as the sensor resolution, sampling density, the expected level of details on the surface as well as the registration error. Although it is possible to parameterize the size of the envelope, this can be addressed by building a multiresolution representation where the envelope size (in numbers of voxels) remains constant.

3. A multiresolution modeling approach

Since the vector field is defined in the neighborhood of the object's surface, most of the voxels in the volumetric grid are unused. It is thus advantageous to avoid allocating those unused voxels by encoding the volume in a hierarchical structure such as an octree. In the octree structure, voxel size divides by two at every level. One can set a maximum resolution level, N, and accumulate surface information in the covariance matrices at the leaf nodes. The octree is then dynamically defined for voxels in the neighborhood of range measurement at level N.

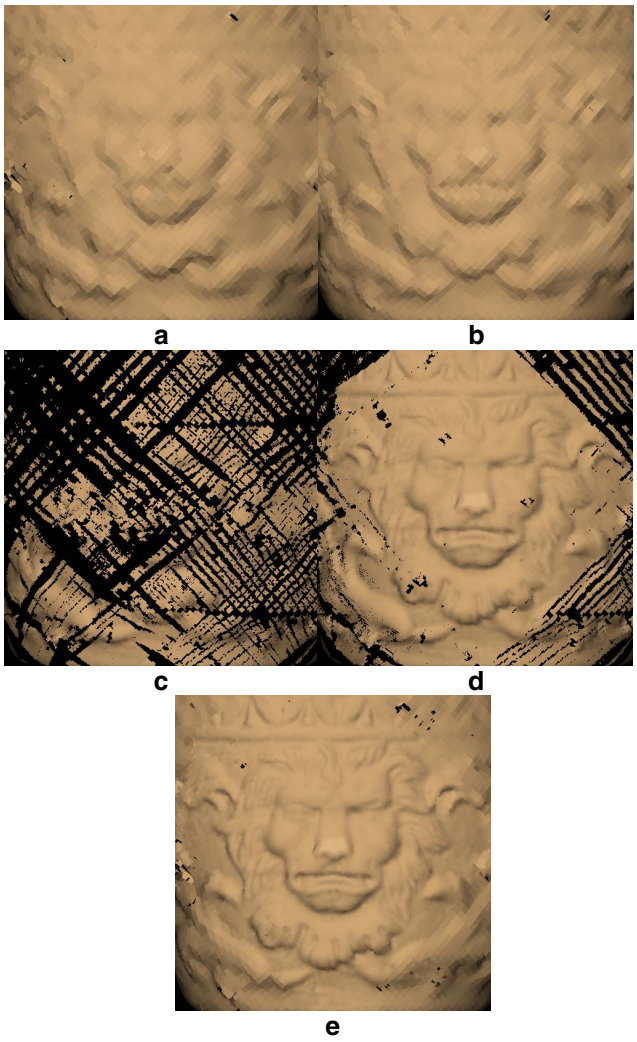

Figure 2. Multiresolution: context and detail with variable sampling density. a) Sparse sampling density at level 7 b) High sampling density at level 7 c) Sparse sampling at level 9 d) High sampling density at level 9 e) Combined view of data at levels 7-8-9 with the highest local level displayed.

Although varying the level of details can be achieved using automatic compression, it is also important to control local resolution and sampling density. In this scenario, a detailed surface section is represented in the context of a larger surrounding area. For this larger area, it is not necessary to capture every detail and use a large quantity of memory (voxels). It is preferable to locally vary the resolution level of the vector field. This can be done manually while the object's surface is being captured. In a typical scenario, a lower resolution level, N, is set and the context is captured in the whole area surrounding and overlapping the region of interest. A higher resolution level $N+P$ is then set and the region of interest is captured at higher sampling density, for instance by slowly moving the hand-guided

sensor within this area. The system is then enabled to build the octree in real-time at higher levels.

In Figure 2, the result after applying this procedure is shown. Figure 2a displays a view of the low resolution section and its surroundings measured at sparse sampling density. In Figure 2b, surface sampling is higher but the resolution of the model, *N*, remains the same. There is no significant difference. Figures 2c and 2d show these two viewpoints built from the same corresponding data, but where the resolution of the model is set to *N+P*. It can be seen that many holes are present in the reconstructed section with lower sampling density since the envelope is also two voxels at the finer resolution. Since the high resolution reconstruction requires high sampling density, low density regions cannot be reconstructed adequately, leaving a large number of holes. The difference thus clearly appears. Figure 2e shows the hybrid representation exploiting both resolutions. Each ray intersects with the volume at its locally higher available resolution level. This causes holes to be filled in the surrounding section while details are visible in the central section. During interactive modeling, one can also set back the resolution level to *N* and only these nodes at level *N* will be updated.

If one changes the resolution level from *N* to *N+P* (with P>1), all intermediate levels are built (independently). In this way, it is possible to maintain the continuity between levels while varying the level of resolution during acquisition. Actually, since the surface envelopes are two voxels with size corresponding to specific levels, the envelope of the vector field at level *K* will generally overlap with the tangent plane of the vector field at level *K-1*. The same surface can thus be tracked in the octree by progressing through the resolution levels. However, this is not strictly guaranteed for any resolution and noise level. In the next section, it will be explained that overlapping through levels is also a necessary property for coherent visualization. Figure 3 shows a 2D cross section of the octree. The overhead and memory requirement for maintaining intermediate levels is not a limitation since the sum of voxels including all levels between *N* and *N+P-1* is below 1/7 of the size at level *N+P*.

One important concern when building and visualizing models is the continuity of the reconstructed surface. Simultaneously displaying at least two levels of resolution may lead to discontinuities at the interface. It is important to understand that the structure encodes the surface at both levels and that there are two models where the sections overlap, as opposed to a mixture of models. It is thus not an objective to hide possible discontinuities since the lower resolution section only presents the context to display the reconstructed detail of interest. Moreover, discontinuities between levels will be minimized when sections overlap over smooth sections. In these areas, a strong discontinuity indicates a misalignment of the data.

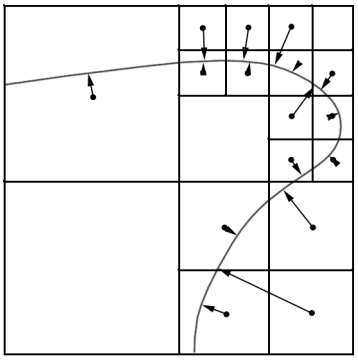

Figure 3. Cross section of the vector field depicting the finest local resolution levels.

4. Interactive visualization

Great efforts have been devoted to accelerating high quality rendering. Pushed by the game industry, the display (rasterization) of triangular meshes, now assisted by texture maps, normal maps and displacement maps to name a few, is a good example. High performance graphics hardware is now widely available. Splatting [14] and its recent versions [10][15] is another worthy method that achieved both fast and fine quality rendering when data can be preprocessed. Finally, pioneering work on efficient ray tracing of volumetric data [6], also followed by recent progress [13], proposed another alternative that can provide renders of the highest quality. Except for quality, ray tracing has not achieved the performance of the former methods especially in the context of interactive modeling. Nevertheless, it solely depends on the number of pixels and octree depth, a property that splatting does not share. When the displayed surface area is small compared with the actual surface area of the model, volumetric ray tracing will be especially indicated. The primary representation for integrating measurements is volumetric, not an explicit surface. Moreover, ray tracing is perfectly tailored to parallelization. For these reasons, our research aims at developing this avenue in the context of interactive modeling, despite current performances of PCs.

4.1 High quality rendering

Integrating rendering *in the modeling loop* allows one to observe the evolution of surface coverage and its quality. It is also possible to observe flaws that are caused by the sensing device. After describing the mechanism of ray tracing in the multiresolution vector field, three methods are compared. They differ in their simplicity (rapidity) and capability to limit discretization and consequently, aliasing along the contour curves of an object.

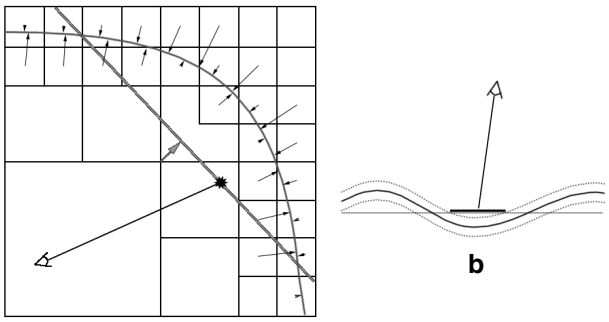

Figure 4. Importance of preserving intermediate levels. a) The tangent plane at level N does not overlap with the envelope at level N+3 at its intersection with the ray path. b) Cross section view: the low resolution level hides the higher surface resolution representation since it is outside the envelope.

The principle of ray tracing is simple. For each pixel in the rendered image, a ray is cast into the volume. When the ray hits the surface, the pixel color (or grey level) is computed from the ray direction, the surface normal orientation and the simulated source direction if different from the ray direction. The depth could also be returned to provide a 2.5D map. The surface is hit at the zero-crossing of the vector field norm that occurs between two neighboring voxels whose vectors have opposite directions within the volume.

The octree is traversed efficiently using the strategy proposed in [7]. However, since the model is multiresolution and the leaf nodes are not all at the same level, we adopt the strategy of displaying the finest local level where available. The ray thus penetrates into the structure at the finest level. This strategy requires that the vector field be defined at each intermediate level between the lower and higher resolution levels. Here again, the envelope of each resolution level, *K*, must overlap with the tangent plane at its adjacent level *K-1*. This can be seen in Figure 4 where the envelope at the higher level does not progressively overlap with the surface representation at the lower level. In this situation, the coarser model occludes the finer level (*N+3*) since there is no active resolution voxel at level *N+1* and the octree is not explored further. The rendered pixel would be displayed based on the coarse level *N*. It is possible to display the appropriate intersection but one would have to explore the sub-tree exhaustively, losing the multiresolution interest. Since the resolution progression is a power of two and the envelope is at least two voxels wide, the same surface will overlap with its adjacent level. Building the intermediate levels makes it possible for each voxel to have a correspondent at coarser levels through a continuous envelope progression.

Three variants of the basic procedure are described; these methods are named “first active voxel”, “first tangent intersection”, and “first entry” respectively.

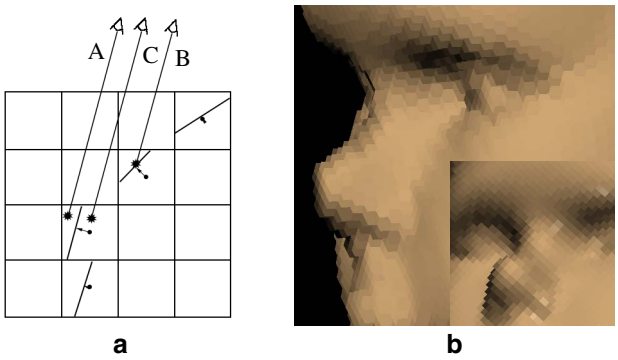

Figure 5. Method “First active voxel”. a) Principle b) Effect on contours and surface.

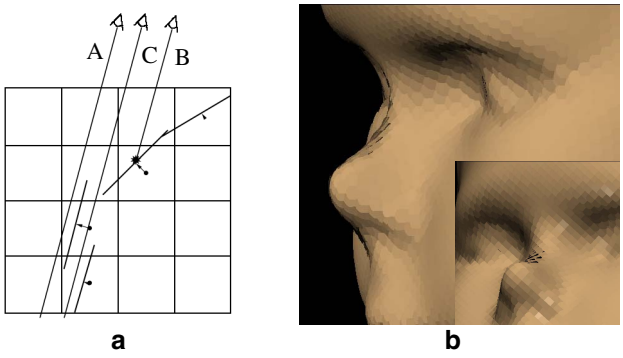

Figure 6. Method “First tangent intersection”. a) Principle b) Effect on contours and surface.

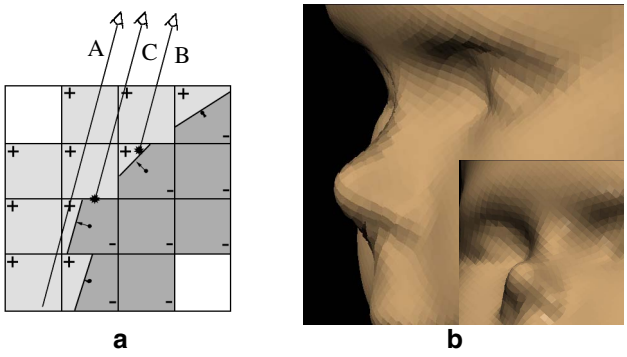

Figure 7. Method “First entry”. a) Principle b) Effect on contours and surface.

The “first active voxel” method detects a hit as soon as the ray penetrates into a voxel within which there is a tangent plane. In practice, this translates to a voxel with a defined normal whose length is less than half the diagonal of the voxel, that is $\sqrt{3}/2$ times the voxel size. Figure 5a illustrates the principle for rays A and B. This approach is rapid and does not lead to hole artefacts (tunnels) due to visualization. The compromise is the stronger discretiza-

tion effect leading to aliasing that is easily observed on occluding contours. The reason for this is clearly seen in Figure 5a where ray A intersects with the voxel but not with the tangent plane of the surface model. One can improve the quality of the rendered image by limiting discretization artefacts. This is accomplished by further imposing that the ray intersects with the tangent plane. In practice, the tangent plane must be bounded to describe a local neighborhood. To do so, it is defined on a circle whose radius length is $\sqrt{3}/2$ times the voxel size. Figures 6a and b show the principle as well as the quality improvement despite a slight increase in the computational cost. Nevertheless, this variant introduces new artefacts that can be seen on the nose of the mannequin head at the lower right part of Figure 6b. These artefacts are caused by rays entering false tunnels into the surface. Ray C in Figure 6a illustrates this all too common situation. Although increasing the circle radius would reduce the problem, a coarser surface of the modeled field would be displayed.

A better solution consists in taking advantage of the volumetric representation where the surface is defined within the envelope as the vector field. This combines the advantages of the two preceding approaches. Actually, each voxel defining the surface can be tagged as inside (sign -), outside (sign +) the surface or both. The two signs are present within voxels split by a tangent plane. The position of the sensor is used to set the signs when the field is built. The ray tracing strategy then consists of searching for sign transition given the sign status of the observer. The interior/exterior boundary may thus be reached 1) *when the tangent plane is crossed over* or 2) *when the ray enters a new voxel.* This approach prevents the appearance of tunnels while limiting aliasing. Figure 7 illustrates the method as well as its resulting effect in b. This last method leads to the best quality results. Nevertheless for the tests we ran, the "First active voxel" method was on average 40% faster while the "First tangent intersection" was about 20% faster.

4.2 Improving speed

Although visualization of the multiresolution representation based on ray tracing is of constant complexity order, it is still the most demanding process in the modeling loop. Depending on the machine technology, a complete visualization can be requested at lower rate than other modeling processes. This leads to a partition of the visualization process where samples of the image to be rendered are displayed at each step. Generally, a dynamic subsampling strategy is necessary for high resolution images with higher acquisition rate. Nevertheless, when the user stops acquiring data to concentrate on the display, we should be able to obtain all of the details. This is the well known principle of progressive displaying.

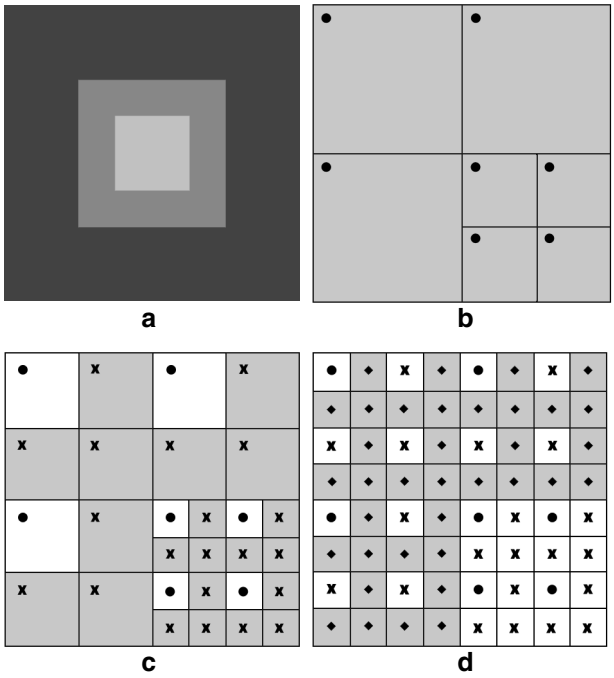

Figure 8. Foveal progression a) Resolution areas from 128-256-512 pixels. In b-c-d, shaded pixels are those updated at the corresponding step. Dots, X and diamonds are reference pixels to select the grey level of the area (2x2 or 1x1 at the finest level). The area displayed in b-c-d is at the upper corner of an area transition.

In this work, we have adopted a foveal sampling strategy that is well suited to hand-guided sensors. For these sensors, the viewpoint can adapt to the camera viewpoint or stay fixed. In both cases, a quick display of details will be prioritized in the center of the image, which is naturally the area of interest. The progressive display is split into 5 iterations. Figure 8 illustrates the progression. In Figure 8a, a *512x512* image is separated into three concentric areas of *128x128, 256x256* and *512x512*. In Figures 8 b, c, and d, the progression is shown at the upper left corner of a transition between two resolution areas. In each figure, the grey shaded areas show the pixels that are updated at a given iteration. The white areas are left at their previous values. The signs (●,✕,◆) mark the pixels that are used to set the uniform grey level in a square block of increasing resolution. One can see that the central area fills more rapidly than its surroundings. Using this scheme, it takes 5 iterations to completely display the *512 x 512* image.

5. Additional results

The additional results presented in this section as well as results presented in the preceding sections were obtained using a hand-held range sensor whose principle was

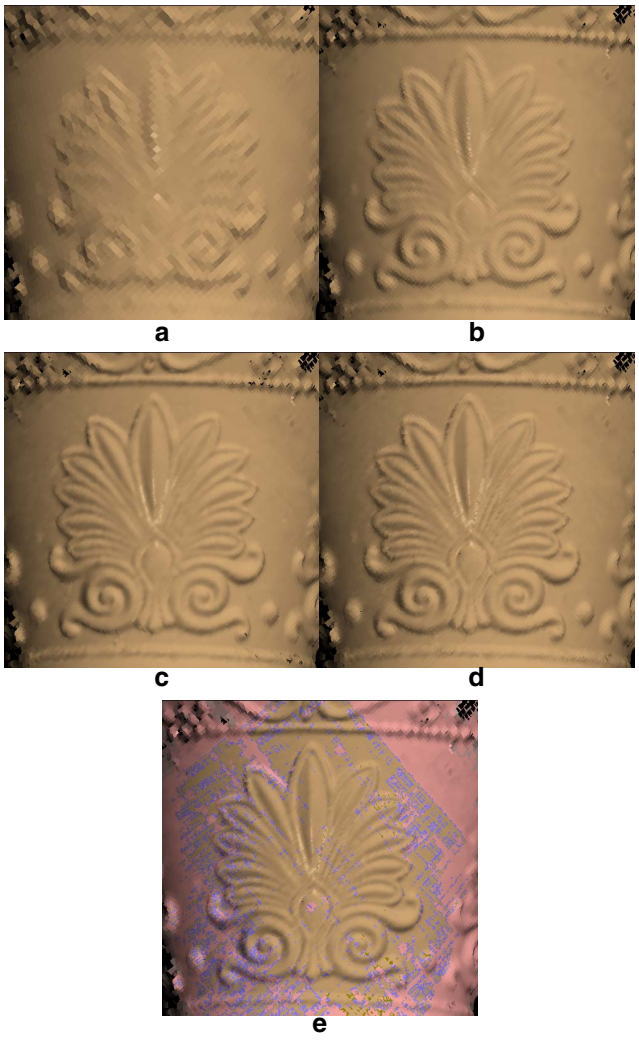

Figure 9. Multiresolution model. In a-b-c-d, the maximum displayed resolution level is limited to 8-9-10 and 11 respectively. e) Same view as in d with 3 colors comprising the coarse, all intermediate and the finest resolution levels.

described in [3]. The sensor projects a crosshair laser pattern. A more recent version of this sensor extracts nearly 18 000 points/sec. at 15Hz. The recent version sensor can also exploit fixed targets for self-referencing on large areas.

Figure 9 shows the levels of the multiresolution representation. The context was captured by setting the voxel size to 5 mm in a cubic volume of 1280 mm for the whole object (a large vase). The 5 mm value corresponds to level 8 in the octree. The object shape was captured quite rapidly without paying attention to a dense sampling of the surface. Then, the resolution level was increased to a voxel size of 0.625 mm which corresponds to the level 11. The central detail displayed in the figure was scanned more

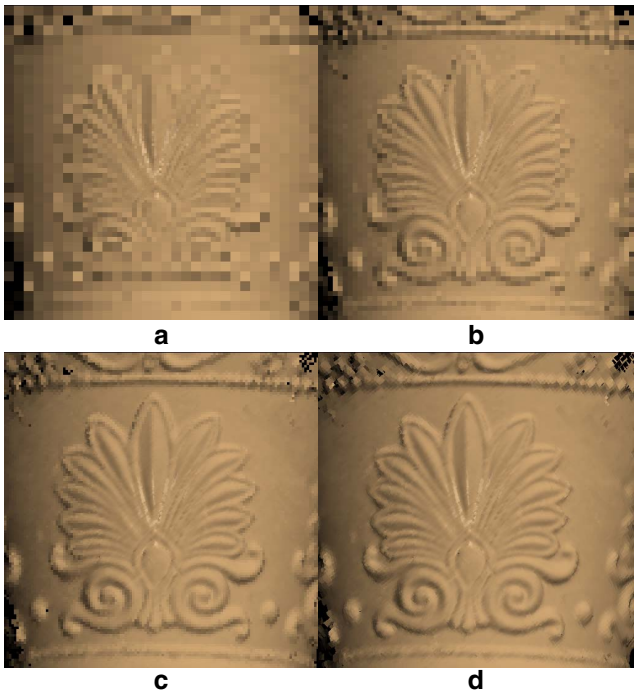

Figure 10. Foveal progression: images. In a-b-c-d, views after steps 1, 2, 3 and 5 are displayed.

carefully. To visualize the multiresolution representation, we have imposed a constraint on the maximum resolution level to display using the "First entry" method. In Figure 9a, b, c, and d, the maximum displayed resolution levels are set to 8, 9, 10 and 11 respectively. We can observe the progression in the details. Figure 9e displays all resolution levels such as in d but with distinct colors for the basis level 8 (pink), the intermediate levels 9 and 10 (blue) and the higher resolution level 11 (beige). It is interesting to observe the blue area that appears at the transition between high and low resolution sampled areas. One property of the system is to capture and display the relevant details adapted to the sampling density. If the sampling density is increased by moving back with the sensor to the same area, the resolution will progressively switch to the finer level.

Figure 10 illustrates the foveal display. The result after the first three steps is displayed in a, b, and c respectively. The image in d corresponds to the last step 5. One can clearly see the variation of progression between the central and the peripheral areas.

For this object with foveal display, time statistics are presented for each of the 5 steps in Figure 11. The curves illustrate the time progression when the higher resolution level varies from 8 to 11. The system was run on a Pentium IV, 3 GHz. Although these specific curves were obtained for the vase object with one view depicted in the upper figures, it clearly illustrates that the progression does not follow the progression in the number of voxels. In theory the

number of surface voxels quadruple at each resolution level since it follows the surface area of the object.

6. Conclusion

Interactive modeling integrates concurrent surface reconstruction and visualization processes. We have exposed the advantages of using vector fields and proposed a multiresolution extension suited for interactive modeling. In this hierarchical representation, leaf nodes are created at different levels, determined during acquisition. This way, it is possible to capture and control the resolution adapted to the level of details and surface sampling. In this framework, the surface model is locally available for all intermediate levels between the finer and the coarser levels. This ensures a better resolution continuity that is exploited for more efficient visualization. Among the three compared methods for visualization, the "First entry" method provides the best quality results.

Future work will concentrate on improving rendering performances that were not highly optimized apart from a foveal progression. In addition, the multiresolution representation is a good tool for exploring more efficient registration approaches.

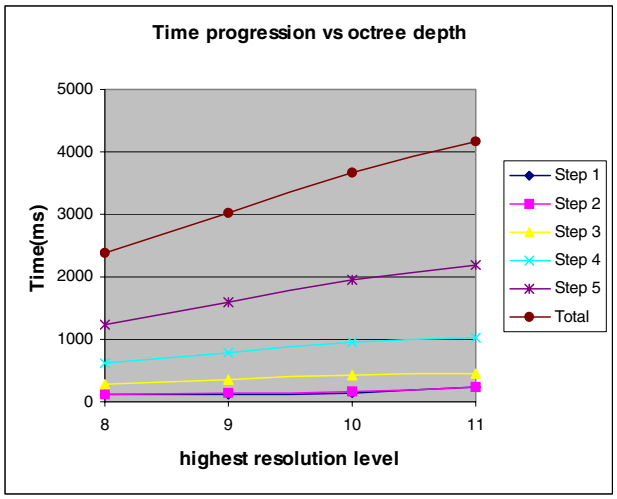

Figure 11. Evolution of rendering time with respect to the highest octree level, for each step of the foveal rendering.

References

[1] T. Bodenmueller and G. Hirzinger, Online Surface Reconstruction from Unorganized 3D-Points for the DLR Hand-guided Scanner System, in *Proc. of the 2nd International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT'04),* Thessaloniki, Greece, Sept. 2004, pp. 175-182.

[2] B. Curless and M. Levoy. A Volumetric Method for Building Complex Models from Range Image, in *SIGGRAPH '96 Proceedings*, 1996, pp. 303-312.

[3] P. Hébert, A Self-Referenced Hand-Held Range Sensor, in *Proc. of the Third International Conference on 3D Digital Imaging and Modeling*, Quebec, Canada, May 2001, pp. 5-12.

[4] A. Hilton and J. Illingworth. Geometric Fusion for a Hand-Held 3D Sensor, *Machine vision and applications*, 12:44-51, 2000.

[5] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, Surface reconstruction from unorganized points, In *SIGGRAPH '92 Proceedings*, volume 26, July 1992, pp. 71-78.

[6] Levoy M., Efficient Ray Tracing of Volume Data, *ACM Transactions on Graphics (TOG)*, Volume 9, Issue 3, July 1990, pp. 245-261.

[7] J. Revelles, C. Ureña, and M. Lastra, An Efficient Parametric Algorithm for Octree Traversal, in *Proc. of the 8th International Conference on Computer Graphics and Visualization'2000*, Plzen (Czech Republic), February 2000.

[8] G. Roth and E. Wibowo. An Efficient Volumetric Method for Building Closed Triangular Meshes from 3-D Image and Point Data. In *Graphics Interface*, May 1997, pp. 173-180.

[9] S. Rusinkiewicz, O. Hall-Holt and M. Levoy, Real-Time 3D Model Acquisition, in *ACM Transactions on Graphics*, 21(3), July 2002, pp. 438-446.

[10] S. Rusinkiewicz and M. Levoy, Qsplat: A Multiresolution Point Rendering System for Large Meshes, in *Proc. of the 27th annual conference on Computer graphics and interactive techniques*, 2000, pp. 343-352.

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

[12] 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 (3DPVT'04),* Thessaloniki, Greece, Sept. 2004, pp. 175-182.

[13] I. Wald, P. Slusallek, State of the Art in Interactive Ray Tracing, in *Proc. of EUROGRAPHICS 2001*, Manchester, United Kingdom, Sept. 2001, pp. 21-42.

[14] L. Westover, Interactive Volume Rendering, in *Proc. of the 1989 Chapel Hill Workshop on Volume Visualization*, Chapel Hill, USA, 1989, pp. 9-16.

[15] M. Zwicker, H. Pfister, J. van Baar, M. Gross, Surface Splatting, in *Proc. of the 28th annual Conference on Computer Graphics and Interactive Techniques*, 2001, pp. 371-378.