Enhanced Vector Quantization for data reduction and filtering

Stefano Ferrari[a,b], Iuri Frosio[a], Vincenzo Piuri[b] and N. Alberto Borghese[a]

a) MAVR laboratory, Department of Computer Science, University of Milano, via Comelico 39/41, 20135 Milano, Italy, [borghese, frosio]@dsi.unimi.it.
b) Department of Information Technologies, University of Milano, via Bramante 65, 26013 Crema, Italy, [ferrari, piuri]@dti.unimi.it.

Abstract

Modern automatic digitizers can sample huge amounts of 3D data points on the object surface in a short time. Point based graphics is becoming a popular framework to reduce the cardinality of these data sets and to filter measurement noise, without having to store in memory and process mesh connectivity.

Main contribution of this paper is the introduction of soft clustering techniques in the field of point clouds processing. In this approach data points are not assigned to a single cluster, but they contribute in the determination of the position of several cluster centres. As a result a better representation of the data is achieved.

In Soft clustering techniques, a data set is represented with a reduced number of points called Reference Vectors (RV), which minimize an adequate error measure. As the position of the RVs is determined by "learning", which can be viewed as an iterative optimization procedure, they are inherently slow. We show here how partitioning the data domain into disjointed regions called hyperboxes (HB), the computation can be localized and the computational time reduced to linear in the number of data points (O(N)), saving more than 75% on real applications with respect to classical soft-VQ solutions, making therefore VQ suitable to the task. The procedure is suitable for a parallel HW implementation, which would lead to a complexity sub-linear in N. An automatic procedure for setting the voxel side and the other parameters can be derived from the data-set analysis.

Results obtained in the reconstruction of faces of both humans and puppets as well as on models from clouds of points made available on the WEB are reported and discussed in comparison with other available methods.

1. Introduction

Volumetric methods are becoming very popular in the pipeline required to contruct a 3D mesh from the clouds of 3D points acquired by modern digitizers [1, 2, 3, 4].

In this approach the volume occupied by the object is partitioned into voxels and all the data points contained inside a voxel are substituted by a single point obtained as a weighted average of these points. This allows reducing the the number of data and filtering the noise on them by local computation only [1, 5, 6]. Moreover, it does not require any additional memory to store and process mesh connectivity [9].

This approach produce also a voxel subdivision of the volume occupied by the scanned object, which is suitable for simple and fast mesh construction algorithms, like marching cubes or piece-wise approximation [7, 8].

Improved versions of clustering algorithms have been recently proposed [20]. In [22, 23] a hierarchical clustering approach is used, which is based on recursively splitting the data set into clusters, which are subdivided until a certain criterion is not met. In the region growing approach, which has been borrowed from the connectionist domain [21], at the beginning, a point is randomly selected and a cluster is grown around it until a certain criterion is met.

Proceedings of the 2nd International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT'04)
0-7695-2223-8/04 $ 20.00 IEEE

Afterwards, a second point is taken and a second cluster is grown around it, by using only the remaining points. The procedure terminates when no data points are left. Both these approaches suffer from the drawbacks offered by hard clustering of the data points: this may lead to sub-optimal solutions with poor visual quality [12, 21]. Morevoer, over fitting and under fitting may occur.

It is shown here how Vector Quantization techniques [11] can be used to obtain a robust solution for volumetric data processing. In this framework, a set of M points, called *Reference Vectors* (RV), is used to represent a set of N > M data points, such that a certain cost function (e.g. reconstruction error) is minimized. Since VQ is a NP-hard problem, sub-optimal solutions are generally accepted. These can be obtained through iterative adaptation of the RVs position [11, 12] and have been developed mainly in the connectionist domain [Baraldi]. They are based on combining soft-max strategies to move the RVs with a deterministic annealing scheduling for the parameters [12, 13]. However, they share two main drawbacks for this application. The annealing-based optimization procedure needs a long time to converge even to a sub-optimal result, and parameter setting is critical.

In this paper it is shown how a regular subdivision of the volume into disjointed regions, Hyperbox (HB), can be used to achieve a large speed up, which makes VQ compatible with mesh construction. Moreover a procedure to determine automatically the optimal voxel side along with the other parameters can be derived [14]. The overall procedure has been termed Enhanced Vector Quantization (EVQ). Results on the reconstruction of 3D digital clones of human faces and puppets from point clouds are reported and discussed.

2. The methodology

In VQ techniques [11], a set of M reference vectors (RV), $W = \{w_j \in R^D\}$, are used to approximate an input set of N data points, $V = \{v_k \in R^D\}$, with N > M. The RVs are distributed such that the following reconstruction error, E(V,W):

$$E(V,W) = \frac{\sum_{k=1}^{N} d\left(v_k, w_j(v_k)\right)^2}{N} = \frac{\sum_{k=1}^{N} \left\| v_k - w_j(v_k) \right\|^2}{N} \qquad (1)$$

is minimized. $w_j(v_k)$ is the RV closest to v_k or "winning" RV.

To determine the optimal position of the $\{w_j(v_k)\}$ RVs, different iterative techniques, based on soft-adaptation, have been proposed mainly in the connectionist domain [12]. At each iteration, t, a sampled point, $\tilde{v}(t)$, is randomly extracted from the input data set and the position of *all* the RVs $\{w_j\}$ is updated according to an updating-rule. Different algorithms differ on their updating rule. Among these, "Neural-Gas" (NG) [17] has proven superior in dealing with outliers, as it is not based on distances but only on ranking, and it will be used here as "computational engine" for VQ.

In NG, the following soft-max rule is adopted to update the RVs position:

$$\Delta w_j(t) = \varepsilon(t) \cdot h_{\lambda(t)}\left(k_j\left(\tilde{v}(t), w_j\right)\right)\left(\tilde{v}(t) - w_j\right) \qquad (2)$$

$k_j \in \{0, 1, 2, ..., M\}$ is the ranking of w_j with respect to the actual data vector $\tilde{v}(t)$, measured according to the Euclidean distance.

$h_{\lambda(t)}(\cdot)$ is the following weighting function:

$$h_{\lambda(t)}(v(t), w_j) = \exp\left(-\frac{k_j(\tilde{v}(t), w_j)}{\lambda(t)}\right) \qquad (3)$$

$\lambda(t)$ controls the number of RVs which are meaningfully updated by the data point $\tilde{v}(t)$. $\lambda(t)$ and $\varepsilon(t)$ decrease as optimization progresses according to:

$$\lambda(t) = \lambda_i\left(\frac{\lambda_f}{\lambda_i}\right)^{t/T_{max}} \qquad (4a)$$

$$\varepsilon(t) = \varepsilon_i\left(\frac{\varepsilon_f}{\varepsilon_i}\right)^{t/T_{max}} \qquad (4b)$$

where T_{max} is the number of iterations. The role of $\lambda(t)$ and $\varepsilon(t)$ is to reduce the number of reference points effectively displaced and the displacement amount induced by $\tilde{v}(t)$ as optimization progresses.

With HB data partitioning, a good distribution of the RVs can be obtained already at start, with a large reduction in the number of iterations and in the computational time.

2.1 Hyper-boxes and RV initialization

In the asymptotic condition, the statistical distribution of the RVs is proportional to that of the data points according to [5, 19]:

$$\rho_w(u) \propto p_v(u)^{\gamma} \quad \text{with} \quad \gamma = \frac{D}{D+2} \qquad (5)$$

where $\rho_w(u)$ and $p_v(u)$ are the density respectively of the RVs and of the data points, and D is the dimensionality of the data space ($\gamma = 0.6$ for D = 3 as

in our case). Let us call τ the desired compression rate:

$$\tau = \frac{M}{N} \quad \Rightarrow \quad M = \tau N \tag{6}$$

and B the region of R^D which contains all the data points. Partitioning B into N_H disjointed regions, $\{B_k\}$, it holds:

$$\bigcup_{k=1}^{N_H} B_k = B \quad \text{with } B_k \cap B_j = \varnothing \quad \forall k \neq j \tag{7a}$$

$$V_B = \sum_{k=1}^{N_H} V_k \tag{7b}$$

where V_B and V_k are the volumes respectively of B and B_k. It results:

$$N = \sum_{k=1}^{N_H} N_k \qquad M = \sum_{k=1}^{N_H} M_k \tag{8}$$

where N_k and M_k are respectively the number of the data points and the RVs inside B_k. Applying (5) to each region B_k, the RVs mean density inside B_k is derived:

$$\bar{\rho}_w(u)\big|_k = \beta \bar{\rho}_v(u)\big|_k^{\gamma} \;\Rightarrow\; \frac{M_k}{V_k} = \beta \left(\frac{N_k}{V_k} \right)^{\gamma} \tag{9}$$

where $\bar{\rho}_w(u)\big|_k$ and $\bar{\rho}_v(u)\big|_k$ are the mean density of the data points and RVs inside B_k. From (7) and (8) follows that:

$$\beta = \frac{M}{\sum_{k=1}^{N_H} N_k^{\gamma} V_k^{1-\gamma}} \tag{10}$$

The number of RVs to be inserted inside each region B_k, M_k, is computed through (6), (9) and (10):

$$M_k = M \frac{N_k^{\gamma} V_k^{1-\gamma}}{\sum_{k=1}^{N_H} N_k^{\gamma} V_k^{1-\gamma}} = \tau N \frac{N_k^{\gamma} V_k^{1-\gamma}}{\sum_{k=1}^{N_H} N_k^{\gamma} V_k^{1-\gamma}} \tag{11}$$

Whenever B is a parallelepiped[1] and the volume of all the sub-regions $\{B_k\}$ is equal, (11) can be simplified as:

$$M_k = M \frac{N_k^{\gamma}}{\sum_{k=1}^{N_H} N_k^{\gamma}} = \tau N \frac{N_k^{\gamma}}{\sum_{k=1}^{N_H} N_k^{\gamma}} \tag{12}$$

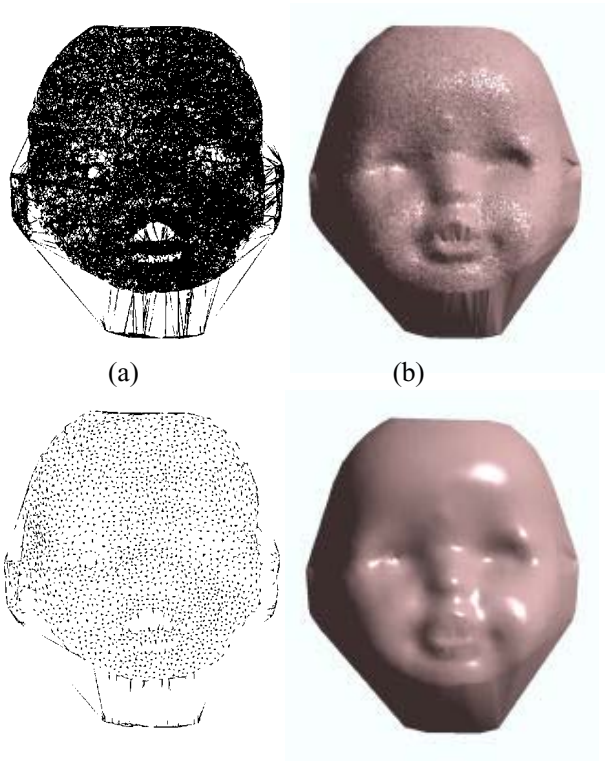
which does not depend on any volume measurement but not on the data points inside each box. Function (12) will be called *partitioning function*.

1 The partitioning schema can be applied, in principle, to boxes of any shape.

The HBs are arranged into a D-dimensional table and the box associated to each data point (or RV) can be directly addressed.

2.2 Speeding-up VQ

HB processing has been used to speed-up the iterative optimization phase in two ways. First, in VQ, $\lambda(t)$ ((3) and (4a)) has to be large at the beginning to allow all the RVs to move through the data space and get to the region of their final destination [15]. Therefore, the first optimization steps are spent to cluster the RVs around the centroid of the data distribution and, only in a second phase, they are distributed towards their final destination. With HB processing instead, this first phase can be skipped as the RVs are placed close to their final position already in the initialization phase.



(a) (b) (c) (d)

Fig.1. The mesh obtained connecting 100,000 points digitized over a puppet face (a). The need of filtering is evident from panel (b). In panels (c-d), the same face is obtained connecting 2,000 RV obtained through EVQ.

A much larger saving in computational time is obtained from the analysis of (2). Here, sorting of all the M RVs is required to compute the ranking k_j[2]. This is an expensive operation, being of the order Mlog(M). However, the utility of displacing the $\{w_j\}$, which lie far from $\tilde{v}(t)$, can be questioned when the w_j position is already close to their optimal one. A better schema is proposed here, where only the RVs inside an *influence region*, $R_\alpha(\tilde{v}(t))$, constituted of the box to which the data point belongs and the adjacent ones is considered. As a result, ordering the RVs does scale anymore with Mlog(M), but it remains constant, equal to: $2^D \overline{M} \log(2^D \overline{M})$.

It should be remarked that if a different compression rate is adopted, the computational time per iteration due to the sorting procedure, does not change. In fact, when the number of RVs increases (larger value of M) L_k decreases and the number of boxes increases consequently such as to keep $\overline{M} \approx M_g$ constant (e.g. compare processing times to obtain Figs.4).

3. Experimental results

This method has been extensively applied to the reconstruction of 3D objects starting from clouds of 3D points sampled over them. A typical data ensemble sampled on a puppet face is reported in Figs. 1a, where a total of N = 100,000 3D points have been taken over the face reported in the bottom right panel. From fig. 1c, where a mesh is obtained connecting the data points, the need of filtering is evident. Moreover, the huge number of triangles obtained (≈ 200,000) calls for data reduction. The method presented here is successful in accomplishing both tasks as can be appreciated in Figs. 1d-1e, where only 2,000 data (≈ 4,000 triangles) are used to represent the same face.

Besides qualitative evaluation, a quantitative evaluation is given. The reconstruction error after 5 N iterations (T_{max} = 500,000) was 1.437 mm^2. The same figure was obtained after only 119,000 iterations (t = 1.19 N) with EVQ. Overall the computational time dropped from 2,660 s on a Pentium II, 350 MHz machine to only 77s. Therefore EVQ allows a large increase in speed and/or accuracy.

Similar results are obtained with different compression rates and smaller data sets like those produce by home-made scanners. In Figs. 2b, 2c and 2d, the mesh obtained with τ = 0.05, 0.1, and 0.15 respectively is reported for the face in Fig. 3a. This is built starting from N = 12,394 sampled points.

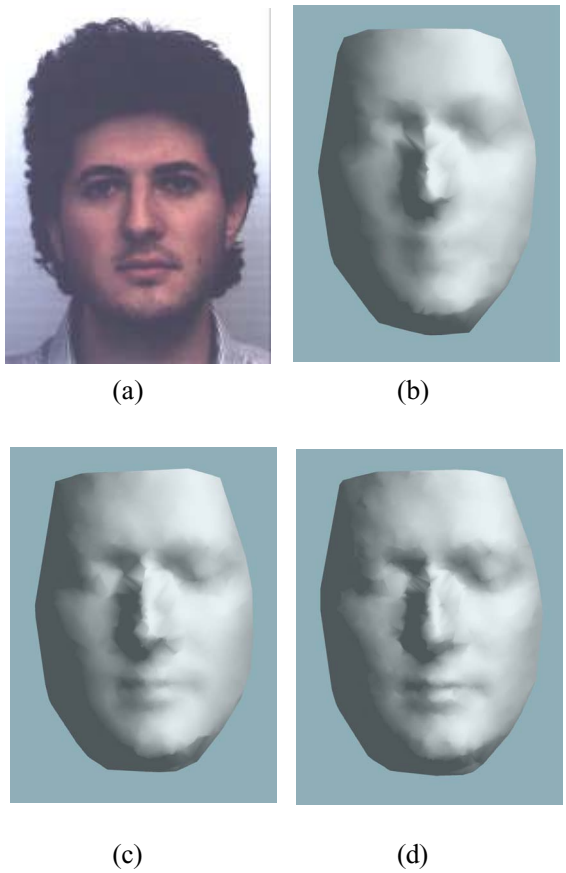


Fig. 2. Mesh constructed from a set of N = 12,394 data points sampled on the face in panel (a) with three different compression rates: (b), τ = 0.05; (c), τ = 0.1; (d), τ = 0.15. The three reconstructions feature 1,213, 2,453 and 3,684 polygons.

HB processing is also particularly suitable to increase the vertex density in selected areas. To the scope only the initialization phase has to be modified. The user has only to define the windows inside which the higher compression rate is required (Fig. 3a-b).

Let us suppose that we want to increase the vertices density for the model in Fig. 3, in the areas of the eyes, the nose and the mouth (Figs. 3a and 3b). First, the number of RVs for each box is computed

[2] Data sorting is required not only by most soft quantization methods [13], but also by some volumetric methods [10].

according to the partitioning function (12) with the lowest compression rate, τ_1. Then, the RVs, which belong to the selected windows, are removed and substituted with a second set of RVs, W_2, such that the desired higher compression rate, τ_2, is achieved in these regions. No modification is required to the RVs outside these windows. Afterwards, the position of the RVs is optimized according to EVQ processing.

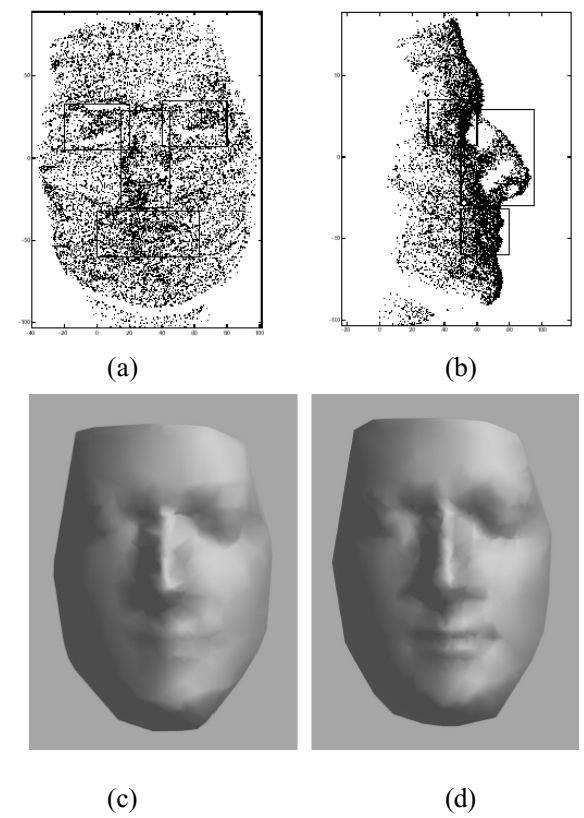


Fig. 3. Multi-rate compression. In panels (a) and (b) the points sampled on the face in Fig. 8a are reported along with the regions inside which the density of the RVs should be increased. The sampled points have been compressed with $\tau_1 = 0.025$ (c) and $\tau_1 = 0.0375$ (d) except inside the selected regions around the mouth, the eyes and the nose where the compression rate is three times τ_1 ($\tau_2 = 0.075$ (c) and $\tau_2 = 0.1125$ (d) respectively). The meshes in panels (c) and (d) are constituted of 941 and 1,594 polygons respectively.

The localized character of EVQ computation forces the RVs inserted inside the higher compression rate windows to stay in the neighborhood. As a result, regions with different compression rates smoothly overlap, but maintain their different RVs density. The results are reported in Figs. 3c-d. In particular the face in Fig. 3c (τ_1 = 0,025 and τ_2 = 0,075) should be compared with that in Fig. 2b and that in Fig. 3d (τ_1 = 0,0375 and τ_2 = 0,1125) with Fig. 2d. The quality of the reconstruction is higher as can be seen from the better outline of the lips, the nose and the eyes. Moreover, this has been obtained using less triangles: only 941 instead of 1,213 in Fig. 4c, and 1,594 versus 2,453 in Fig. 4d.

Similar results have been obtained with data made accessible on the WEB and in particular by the Stanford Computer Graphics Laboratory repository [16]. Results on the dragon model (Fig. 4a), constituted of 437,645 data points, are reported in Fig. 4c-d. As expected the detail increases with the number of RVs, although the visual quality is already close to the original with a compression rate as low as the 10%. As discussed in Section 3., processing time is almost independent on the number of RVs thanks to the partitioning schema adopted, and it was measured as 3-4 minutes for a Pentium III, 1 Ghz, 320Mbyte RAM, machine (see caption of Figures 4).

4. Discussion.

The reconstruction of a 3D mesh starting from a set of 3D points is a complex process, which requires both noise elimination and data reduction. This process is carried out usually on very large data sets which makes the process extremely slow. EVQ is introduced here as a pre-processing stage aimed to reduce the number of points and reduce the measurement noise in a reasonable amount of time.

4.1 Comparison with other approaches

The method proposed here can be applied successfully in the field, which has been recently termed "Point-based graphics" [17], where points sampled over a mani-fold are processed without any implicit or explicit information on the mani-fold itself.

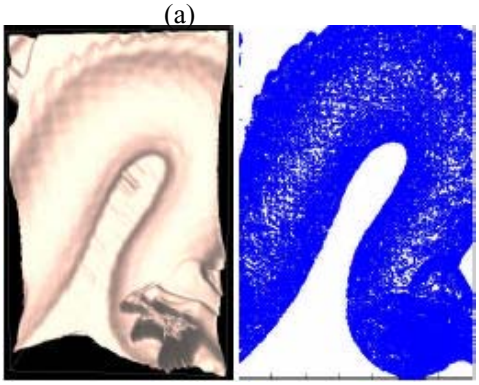
Main contribution of this paper is the introduction of soft clustering techniques to the field of point based graphics. These techniques do not associate a data point to a unique cluster, but a single data point does contribute to define the position of several cluster centers.

This, in combination with a disjoint partitioning of the data space into macro voxels,

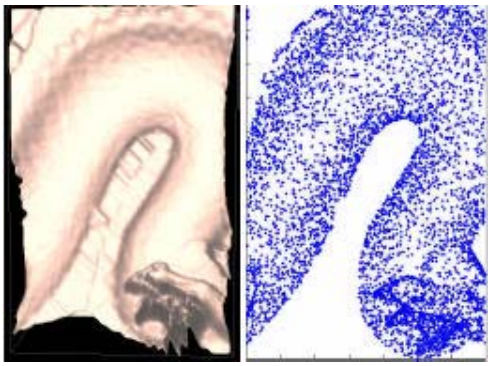
called Hyper box processing, allows overcoming the problems encountered by the hard clustering procedures used in the field up to now [1, 6, 19, 20, 22, 23].

(a)

(b)



(c)



(d)

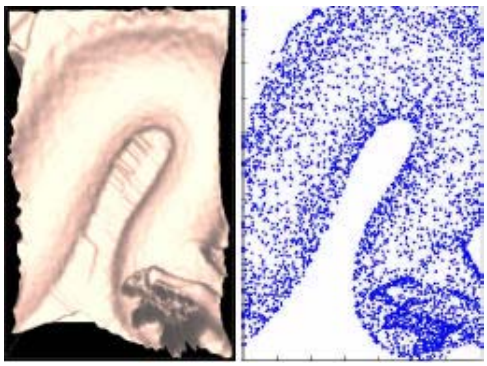


Fig. 4. The dragon data set [16] is constituted of 437,645 points (a). A close look to a detail of a paw and of the body is reported in panel (b). This same detail has been taken from the dragon compressed at τ = 10% 44,202 RVs were generated in 275,8s, (c) and 21,882 RVs were generated in 262,18s, τ = 5% (d).

Similarly to here, a partitioning of the data space was also carried out at start in [1, 6]. However, the scope of the partition was different from here. As a single cluster center was generated for each voxel, voxels have to be very small; for instance a voxel side of 0.5mm, which corresponds to half the range image sampling, was suggested in [1] (for sake of comparison, a voxel side of 12.2mm was defined for the data in Fig. 1). This produces a regular spacing of the cluster centers, which does not reflect the local differential properties of the mani-fold. Other drawbacks induced by hard clustering procedures are surface thickening [1, 6] and spurious discontinuities [12], which are due to assigning close points to two different (adjacent) clusters. This enhances measurement noise as in the mesh construction phase it produces a spurious local peaks and gradients on the surface.

In EVQ each box contains several cluster centers, which are positioned such as to represent the data locally that is those data, which lie inside their Influence Region. As Influence Regions partially overlap, thickening and surface discontinuities do not occur. Moreover, the error function (1) forces more RVs in those regions where surface is more variable ($|v_k - w_j|$ is larger), producing a denser set of RVs is the most variable regions (cf. Fig. 1d).

4.2 Computational time

It can be shown [14] that the computational cost of EVQ is of the order $O(\overline{M} \log \overline{M})$, a small fixed quantity, independent of the total number of the RVs. In fact, when the number of RVs is increased, the box side is automatically decreases, leaving the cost unchanged. We remark explicitly that the cost does not depend on the total number of data points. Experimentally this produces a speed-up of more than five times with respect to standard NG. This was consistently observed on data sets of different sizes, with different compression rates.

The nice property of EVQ is the possibility of deriving a fully parallel implementation. This would lead to a complexity sub-linear in the number of RVs and could open the door to real-time implementation. This property is shared with those hard clustering techniques, which are based on a disjointed partitioning of the data space, achieved both at start [1, 6], or incrementally, through a progressive octree subdivision, as clustering progresses [24],

The drawback of voxel based data processing is extra-memory occupancy, which was acknowledged to be a large problem for those approaches based on micro-voxels [1, 6]. In fact, efficient voxel partitioning would require a table to immediately access to the data. This requires one pointer per box, which produces linear increase of the memory occupancy, with the number of boxes per dimension and exponentially with the space dimension, D. Therefore solutions which are a trade-off between memory occupancy and algorithm speed have been recently proposed [1, 5, 19]. In these approaches, to avoid the waste of memory constituted of the allocation for those voxels which do not contain sampled points, a list of the not empty voxels has been substituted to the table. However, the determination of the box associated to a given sampled point becomes a search in a hash table, whose computational cost increases linearly with the number of HBs.

This problem is not as dramatic in HB processing, as macro voxels are used: memory occupancy of the HB pointers is much smaller than the dimension of the data. For example, a total of 1,440 boxes were used for the data in Fig. 1, with memory occupancy of 5,760 byte, and 104,811 for the data in Fig. 4 (with a compression rate of 50%[3]), with memory occupancy of 420kbyte. Therefore the more efficient table structure has been adopted here.

5. Conclusion

An improved soft clustering technique combined with macro-voxel data processing has been presented here (EVQ). It allows reducing the cardinality of the data clouds produced by modern 3D scanners efficiently by eliminating measurement noise. The use of soft clustering allows avoiding thickening and surface discontinuities which may be obtained by hard clustering techniques deployed in the field of point based graphics.

[3] For sake of completeness, 109,120Kbyte for $\tau = 20\%$ and only 14,400Kbyte for $\tau = 5\%$.

Acknowledgements

This work was partially supported by MIUR GRANT 2003011918_004.

References

[1] S. Rusinkiewicz, O. Hall-Holt and M. Levoy, Real-time 3D Model Acquisition, *Proc. Siggraph'02*, ACM Press, pp. 438-446.

[2] C. Stretcha and L. Van Gool, Dense matching of multiple wide-baseline views. *Proc. ICCV'03*.

[3] N. D'Apuzzo, *Surface Measurement and Tracking of Human Body Parts,* PhD Thesis, N.81, Institute of Geodesie and Photogrammetry, ETH, Zurich, 2003.

[4] M. Levoy, S. Rusinkiewicz, M. Ginzton, J. Ginsberg, K. Pulli, D. Koller, S. Anderson, J. Shade, B. Curless, L. Pereira, J. Davis and D. Fulk, "The Digital Michelangelo Project: 3D Scanning of Large Statues," *Proc. Siggraph'99*, ACM Press, pp. 121-132, 1999.

[5] G.Roth and E.Wibowoo, 1997, An efficient Volumentric Method for Building Closed Triangular Meshes from 3D image and Point Data, *Proc. Graphics Interfaces,* 1997.

[6] B.Curless and M.Levoy, 1996, A volumetric method for building complex models from range images, *Proc. Siggraph'96,* ACM Press, pp. 303-312.

[7] Lorensen, W.E. and Cline, H.E., "Marching Cubes: a high resolution 3D surface reconstruction algorithm," *Computer Graphics,* Vol. 21, No. 4, pp 163-169 (*Proc. of SIGGRAPH*), 1987.

[8] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, W. Stuetzle, "Piecewise Smooth Surface Reconstruction", *Proc. Siggraph'94*, ACM Press, pp. 295-302. 1994.

[9] M.Isenburg and S.Gumhold**,** Out-of-Core Compression for Gigantic Polygon Meshes, *Proc. Siggraph'03,* ACM Press, pp. 935-942.

[10] K.L. Low and T.S. Tan, "Model Simplification Using Vertex Clustering," *ACM Symposium on Interactive 3D Graphics*, pp. 75-81, 1997.

[11] T. Hofman and J.M. Buhman, "Competitive Learning Algorithms for Robust Vector Quantization," *IEEE Trans. on Signal Proc.,* vol. 46, no. 6, pp. 1665-1675, 1998.

[11] Gersho and R.M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Publisher, 1995.

[12] A. Baraldi, P. Blonda, "A survey of fuzzy clustering algorithms for pattern recognition - Part I," *IEEE Trans. on System, Man and Cybernetics, Part B: Cybernetics*, vol. 29, no. 6, pp.778-785, Dec. 1999.

[13] T.M. Martinetz, G. Berkovich, K.J. Schulten, "Neural-Gas Network for Vector Quantization and its Application to Times-Series Prediction," *IEEE Trans. on Neural Networks*, vol. 4, no. 4, pp. 558-569, 1993.

[14] S. Ferrari, *Una Metodologia Gerarchica per Analisi e Sintesi di Immagini Tridimensionali* . PhD thesis, Politecnico di Milano, Jan. 2001.

[15] C. Darken and J. Moody, "Note on learning rate schedules for stochastic optimization," *Adv. Neural Inform. Process. Syst.*, vol. 3, 1991.

[16] URL: http://www-graphics.stanford.edu/ data/3Dscanrep/.

[17] M.Pauly and M.Gross, Spectral Processing of Point-Sampled Geometry, Proc. *Siggraph'01,* ACM Press, pp. 211-219.

[18] C.I. Connolly, Cumulative generation of octree models from range data. *Proc. Intl. Conf. Robotics,* pp. 25-32, 1984.

[19] J. Rossignac and P. Borrel, "Multi-Resolution 3D Approximation for Rendering complex Scenes," *Modelling in Computer Graphics*, B. Falcidieno and T. L. Kunii, Eds. Springer-Verlag, pp. 455-465, 1993.

[20] M. Pauly, H.M. Gross, L. Kobblet, 2002, Efficient Simplification of Point-Sampled Surfaces, *Proc. Visualization, 2002*.

[21] B. Fritzke, Growing cell structures - a self-organizing network for unsupervised and supervised learning, *Neural Networks* (1994), Vol. 7, No. 9, page 1441-1460.

[22] D. Brodsky, D. Watson, 2000, Model simplification through refinement, *Proc. of Graphical interfaces, 2000*.

[23] E. Shaffer, M. Garland, 2001, Efficient Adaptive Simplification of Massive Meshes, *Proc. IEEE Visualization 2001*.