2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

3D Colour Object Reconstruction based on Growing Neural Gas

Sergio Orts-Escolano Jose Garcia-Rodriguez Vicente Morell Miguel Cazorla Juan Manuel Garcia-Chamizo

***Abstract*— With the advent of low-cost 3D sensors and 3D printers, surface reconstruction has become an important research topic in the last years. In this work, we propose an automatic method for 3D surface reconstruction from raw unorganized point clouds acquired using low-cost sensors. We have modified the Growing Neural Gas (GNG) network, which is a suitable model because of its flexibility, rapid adaptation and excellent quality of representation, to perform 3D surface reconstruction of different real-world objects. Some improvements have been made on the original algorithm considering colour information during the learning stage and creating complete triangular meshes instead of basic wire-frame representations. The proposed method is able to create 3D faces online, whereas existing 3D reconstruction methods based on Self-Organizing Maps (SOMs) required post-processing steps to close gaps and holes produced during the 3D reconstruction process. Performed experiments validated how the proposed method improves existing techniques removing post-processing steps and including colour information in the final triangular mesh.**

I. Introduction

Many well established techniques proposed solutions to the 3D representation and surface reconstruction problem from a geometric point of view. However, these algorithms required long times to process the input point cloud and do not scale properly with very large data [1], [2]. Moreover, these traditional geometric approaches do not manage non-stationary distributions and do not deal with the lack of a priori information about the input space, e.g. the presence of multiple shapes in the point cloud and noise induced by the sensors.

Considering the 3D representation problem from a computational intelligence approach and based on self-organization maps, a different perspective to obtain 3D reconstructions is proposed. These methods could be considered as flexible and growing models . Moreover, we can find some similarities or correspondences between the neural network map and the 3D representation obtained. Nodes of the neural network map correspond to vertices of a mesh and connection between nodes correspond to the edges. Therefore, in this work the terms node and vertex, and connection and edge are used interchangeably. From this perspective, some methods were proposed based on self-organizing maps.

In [3] it is proposed the use of Kohonen's self-organizing map for surface reconstruction using as an input data unorganized point clouds. Moreover, since Kohonen's map does not produce regular faces, an edge collapse operation was introduced eliminating dangling faces. This approach presented some drawbacks as if the real object surface has concave structures, applying Kohonen's learning algorithm has some difficulties to correctly approximate those parts. In addition, as the Kohonen's algorithm has a high computational cost, the single thread CPU implementation presented in this work took more than one hour to represent the Stanford bunny model. Presented method was only tested with synthetic data and the bunny model, which is comprised of $34,834$ points. Junior et al. [4], extended [3] introducing new mesh operators that allowed it to perform improvements on the surface geometry: edge swap, edge collapse, vertex split and triangle subdivision. Moreover, the method introduced a new step to remove unstable vertices using the mean distance and the standard deviation of the 3D representation regarding the sampled input space. Although this new approach improved the surface geometry, the method does not deal with concave or non-convex regions and the initial structure of the representation has to be pre-established considering the topology of the input space. The fixed structure of the SOM does not learn the spatial relationships between the vertices and therefore does not generate a model that accurately represents the shape of the input space. To overcome this problem, some methods based on Growing SOMs were proposed.

One of these SOM-based methods is the Growing Cell Structures (GCS) algorithm [5], which is a model formed incrementally. However, it constraints the connections between the nodes, so any model produced during training is always topologically equivalent to the initial topology. In [6] it is used the GCS algorithm to reconstruct objects surface. Meshes operators are used to change the connectivity of the mesh and therefore final topology is always equivalent to the initial mesh.

The Topology Representing Networks (TRN), proposed by [7], does not have a fixed structure and also does not impose any constraint about the connection between the nodes. In contrast, this network has a pre-established number of nodes, and therefore, it is not able to generate models with different resolutions. The algorithm was also coined with the term Neural Gas (NG) due to the dynamics of the feature vectors during the adaptation process, which distributes themselves like a gas within the data space. Barhak [8] proposed a NG-based surface reconstruction algorithm since this network has the ability to accurately represent the topology of a point cloud. However, as the NG has a fixed number of nodes, it is necessary to have some a priori information about the input space to pre-establish the size of the network. This model was

Sergio Orts-Escolano, Jose Garcia-Rodriguez and Juan Manuel Garcia-Chamizo are with the Department of Computer Technology of the University of Alicante (email: {sorts, jgarcia, juanma}@dtic.ua.es).

Vicente Morell and Miguel Cazorla are with the Computer Science and Artificial Intelligence Department of the University of Alicante (email: {vmorell, miguel}@dccia.ua.es).

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

extended by [9] proposing the Growing Neural Gas (GNG) network, which combined the flexible structure of the NG with a growing strategy. Moreover, the learning adaptation step was slightly modified. This extension enabled the neural network to use already detected topological information while training in order to conform to the geometry. This approach has the capability to add neurons while preserving the detected topology.

As the original GNG algorithm does not produce faces and the generated map is a wire-frame representation model, some works extended the original algorithm to produce faces. In [10], the GNG network is employed to model a point cloud and those regions that need further sampling in order to obtain a more accurate model. Rescanning at higher resolution is performed for each identified region of interest and a multi-resolution model is built. In this work, only nodes of the generated map are used as the work is focused on sampling capabilities of the GNG. MGNG [11] applied some postprocessing steps in order to perform surface reconstruction once the map is generated using the original GNG algorithm. Most of these approaches were tested against CAD models or synthetic data and only few experiments were performed on objects acquired with 3D sensors. In [12], the GNG algorithm was modified in order to produce topological faces. The extended method was called Growing Self-Reconstruction Maps (GSRM) and some learning steps as CHL and the operation of vertex insertion and removal were also modified. Most experiments of this work were performed on the Stanford dataset, which had been previously filtered and therefore the surface reconstruction step does not have to deal with noisy input spaces produced by common 3D sensors. In [12], [8] the Competitive Hebbian Learning was extended considering the creation of 2-manifold meshes and face reconstruction. However, it was also required to apply some post-processing steps to create a complete model.

Although the use of the SOM-based techniques as NG, GCS or GNG for 3D input space representation and surface reconstruction has already been studied and successful results have been reported, there are some limitations that still persist. Most of these works assumed noise-free point clouds. Therefore, applying these methods on challenging real-world data obtained using noisy 3D sensors have not been object of study yet. Moreover, with the advent of low cost RGB-D cameras as the Microsoft Kinect [1] partial point clouds have to be considered. Besides providing 3D information, these devices also provide colour information, feature that was not considered in the revised works.

The rest of the paper is organized as follows: first, a section describing briefly the GNG algorithm is presented. In Section III the modification of the GNG algorithm, considering colour information during the learning process, is detailed. Section IV presents and depicts the proposed algorithm for face creation during the network learning process. In Section V we present some experiments and discuss results obtained using our novel approach compared to existing methods. Finally, in section VI we give our conclusions and directions for future work.

[1]Kinect for XBox 360: http://www.xbox.com/kinect Microsoft

II. GNG Algorithm

With Growing Neural Gas (GNG) [9] method a growth process takes place from minimal network size and new neurons are inserted successively using a particular type of vector quantization. To determine where to insert new neurons, local error measures are gathered during the adaptation process and each new neuron is inserted close to the neuron with highest accumulated error. At each adaptation step a connection between the winner and the second-nearest neuron is created as dictated by the Competitive Hebbian Learning (CHL) algorithm. This is continued until an ending condition is fulfilled, as for example evaluation of the optimal network topology or a fixed number of neurons is reached. The network is specified as:

- A set N of nodes (neurons). Each neuron $c \in N$ has its associated reference vector $w_c \in R^d$. The reference vectors can be regarded as positions in the input space of their corresponding neurons.
- A set of edges (connections) between pairs of neurons. These connections are not weighted and its purpose is to define the topological structure. An edge aging scheme is used to remove connections that are invalid due to the motion of the neuron during the adaptation process.

The GNG learning algorithm is as follows:

1) Start with two neurons a and b at random positions w_a and w_b in R^d.
2) Generate at random an input pattern ξ according to the data distribution $P(\xi)$ of each input pattern.
3) Find the nearest neuron (winner neuron) s_1 and the second nearest s_2.
4) Increase the age of all the edges emanating from s_1.
5) Add the squared distance between the input signal and the winner neuron to a counter error of s_1 such as:

$$\triangle error(s_1) = \|w_{s_1} - \xi\|^2 \tag{1}$$

6) Move the winner neuron s_1 and its topological neighbors (neurons connected to s_1) towards ξ by a learning step ϵ_w and ϵ_n, respectively, of the total distance:

$$\triangle w_{s_1} = \epsilon_w(\xi - w_{s_1}) \tag{2}$$

$$\triangle w_{s_n} = \epsilon_n(\xi - w_{s_n}) \tag{3}$$

   For all direct neighbors n of s_1.
7) If s_1 and s_2 are connected by an edge, set the age of this edge to 0. If it does not exist, create it.
8) Remove the edges larger than a_{max} . If this results in isolated neurons (without emanating edges), remove them as well.
9) Every certain number λ of input patterns generated, insert a new neuron as follows:
   - Determine the neuron q with the maximum accumulated error.

- Insert a new neuron r between q and its further neighbor f:

$$w_r = 0.5(w_q + w_f) \tag{4}$$

- Insert new edges connecting the neuron r with neurons q and f, removing the old edge between q and f.

10) Decrease the error variables of neurons q and f multiplying them with a consistent α. Initialize the error variable of r with the new value of the error variable of q and f.
11) Decrease all error variables by multiplying them with a constant γ.
12) If the stopping criterion is not yet achieved (in our case the stopping criterion is the number of neurons), go to step 2.

III. Colour interpolation

As modern 3D sensors provide colour information, the proposed method was modified regarding the original version considering also point cloud colour. Input space dimension is increased from 3 to 6 adding red, green and blue colour components. Now the input distribution is defined in $\mathbb{R}^d$ where $d = 6$. Most SOM-based approaches already presented only considered spatial information as neuron's weight vector w_c, so we modified the learning algorithm adding colour to the neuron's weight vector w_c and considering it during the learning process, now the dimension of the neuron's weight vector is 6 including spatial and colour information. Color values were normalized ranging from 0.0 and 1.0. Colour information is considered during the weight adaptation process but it was not included in the CHL (winning neurons) stage as we still are focused on preserving the topology of the input space. Therefore, winning neuron stage only compute Euclidean distance using x,y,z components. Figure 1 shows how the GNG method generated a down-sampled version of captured coloured point clouds, interpolating the colour of original observations and achieving a good topological fitting for different objects and scenes. We called this version Colour-GNG.

In order to validate and compare the colour version of the GNG, we implemented a different strategy to consider point cloud colour. Instead of adding colour information to the learning process, a post-processing step to compute colour information is added to the process. Once the network has been adapted to the input space (original GNG) and it has completed the learning process, each neuron of the network computes colour information from closest input patterns. Colour information of each neuron is calculated as the average of weighted values of the K-nearest input patterns, obtaining an interpolated value of the surrounding point. Colour values are weighted using Euclidean distance from input pattern to its closest neuron reference vector. K-nearest neighbours are obtained using a radius search process, using as a radius the resolution of the generated map by the GNG algorithm. Therefore, RGB colour for each neuron is calculated using the following equation:

Fig. 1. Different objects are down-sampled using the Colour-GNG representation. (a),(b) show original pointclouds. (c),(d) show down-sampled point clouds using the proposed method.

$$RGB_i = \psi \sum_{\forall j \in N_i} (RGB_j \cdot w(j-i)) \tag{5}$$

where N_i represents the nearest input patterns of the neuron i, i is the neuron being processed and $w(j-i)$ is the distance weighted function between the neighbouring pattern j and the neuron itself. Using that distance weighted function the weight of the colour of the input pattern decays exponentially as the distance to the neuron increases. ψ is a normalization factor that makes RGB components range between 0.0 and 1.0.

$$w(j-i) = e^{-\|j-i\|} \tag{6}$$

Figure 2 visually shows this process. Although this search is accelerated using a Kd-tree structure it is considerably slower than the colour version of the GNG. Colour-GNG in the same learning process is able to adapt its neurons' weights fitting accurately to the input space.

Figures 3 and 4 show various observations that have been created using both approaches. Colour-GNG produces a map that successfully interpolates input colour information producing a useful down-sampled map. Moreover, results were similar to those obtained with the colour interpolation post-processing step.

Finally, some quantitative results are presented in Table I showing the mean error between estimated colours using the post-processing interpolation step and the proposed Colour-GNG method. The error is computed over the three

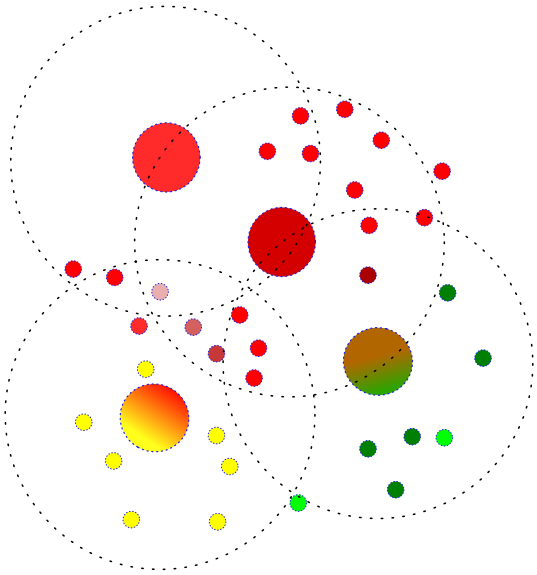

Fig. 2. Colour interpolation. Colour assigned to each neuron (large circles) is calculated as the averaged weighted sum of input space samples (small circles) within a search radius (dotted circle).

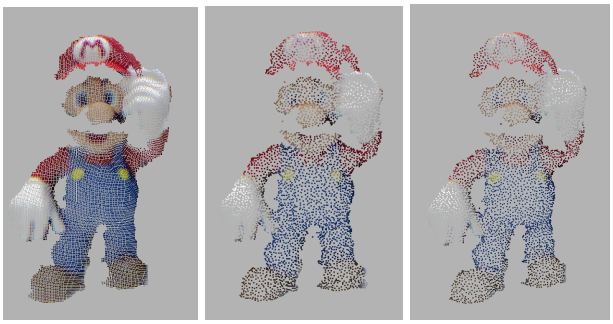

Fig. 3. Mario figure is down-sampled using the Colour-GNG method. Results are similar to those obtained with the colour interpolation post-processing step.

components of the RGB model. The maximum error obtained in a component between estimated colours using the post-processing step and the Colour-GNG is less than four units, considering that each component is represented using unsigned char format. In this experiment we used values between 0 and 255 to facilitate understanding of the obtained results.

With these results we can conclude that Colour-GNG method is able to obtain similar results compared to complex post-processing steps and reducing the processing time.

IV. 3D SURFACE RECONSTRUCTION

Figure 5 shows the result of using an existing GNG-based method for surface reconstruction [12] without applying post-processing steps. The reconstructed model has a lot of gaps and holes that makes the model not suitable for computer vision applications.

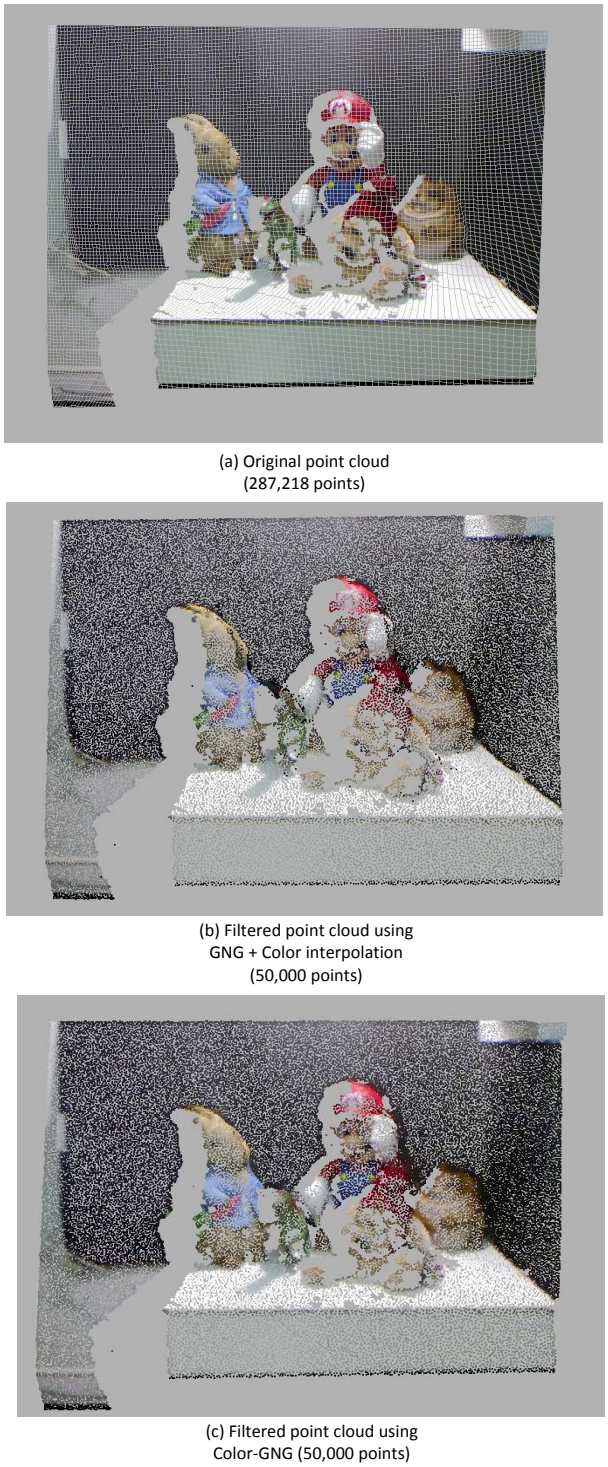

Fig. 4. Two different scenes captured using the Kinect sensor are represented using the Colour-GNG method. Results are similar to those obtained with the colour interpolation post-processing step.

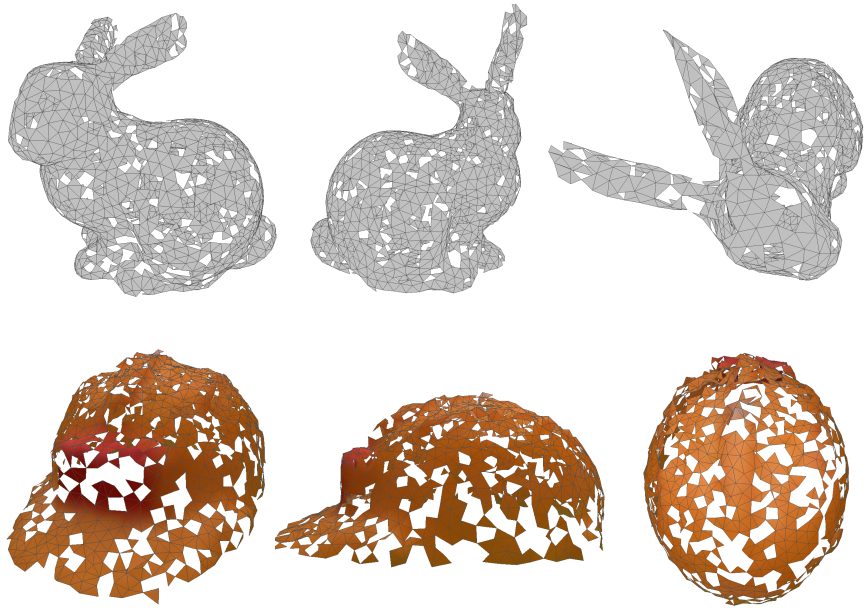

Fig. 5. Different views of reconstructed models using an existing GNG-based method for surface reconstruction. Post-processing steps were avoided causing gaps and holes in the final 3D reconstructed models.

TABLE I

COLOUR MEAN ERROR BETWEEN COMPUTED COLOURS USING A POST-PROCESSING INTERPOLATION STEP AND THE COLOUR-GNG.

	Mean error		
	Red	**Green**	**Blue**
Scene 1 - $20,000n$ 50λ	1	1	2
Scene 1 - $50,000n$ 100λ	1	1	1
Scene 2 - $20,000n$ 50λ	4	4	4
Scene 2 - $50,000n$ 100λ	2	2	2
Object 1 - $3,000n$ 50λ	2	3	3
Object 1 - $5,000n$ 100λ	1	2	2
Object 2 - $3,000m$ 50λ	3	3	3
Object 2 - $5,000n$ 100λ	2	2	2

In this section, we detail the proposed extension of the already described GNG method to generate full coloured 3D models without applying post-processing steps.

A. Extended CHL

Original CHL, presented in Section II, only considered the creation of edges between neurons producing wire-frame 3D representations. Therefore, it is necessary to modify this process in order to create triangular faces during the learning process. Based on [12] and [8] we extended the CHL developing a method able to produce full 3D meshes. In contrast to existing methods mentioned above, our extension does not need post-processing steps. The 3D mesh is created during the learning stage.

The edge creation stage, was also extended considering the creation of triangular faces during this process. Algorithm 1 describes our extended CHL to produce triangular faces.

In order to avoid non-manifold and overlapping edges, the edge creation step was modified restricting the creation of edges if the winning neurons s_1 and s_2 have already more than two common neighbours. This constraint helps avoiding edges with more than two incident triangles. Then, for every sampled point, a face is created whenever the already existing edges or the new ones form a triangle. Moreover, if the creation of faces would produce edges with more than two incident faces, then the face is not created avoiding overlapped triangles and non manifold meshes. During the creation of triangular faces it is checked if the face to be created already exist, in that case, the face is not created. Figure 6 shows common situations produced during the CHL and how our method create edges and triangular faces in those cases.

The age scheme presented in the original GNG algorithm was also considered to remove those edges that have an age higher than a given threshold age_{max}. This age scheme was extended including the removal of faces that shared this edge. Furthermore, to obtain regular triangular faces we included another constraint that was introduced in [13]. This constraint is based on the Thales sphere concept. For every edge that already existed in the CHL process, this mechanism computes the angle between the vectors formed by $s_1 - s_2$ and $s_1 - n_i$ where n_i is a common neighbour of s_1 and s_2. If this angle $\theta > \theta_{max}$ the edge between s_1 and n_i is removed. Faces incident to this edge are also removed. Different values for θ_{max} were tested, obtaining regular triangles for θ_{max} values between $2/3\pi$ and $3/4\pi$. Figure 7 shows this process.

input : A point cloud
output: 3D mesh

For each input pattern presented to the network, the two nearest neurons to the input pattern are selected as winning neurons s_1 and s_2;
if *s_1 and s_2 are already connected by an edge* **then**
  Set edge age to 0 in order to "reinforce" it;
  Check edge removal mechanism based on the Tales Sphere;
  if *s_1 and s_2 have one or two common neighbours* **then**
    foreach *common neighbour n_i* **do**
      Create a face f using s_1, s_2 and n_i;
    end
  end
  if *s_1 and s_2 have one or less common neighbours* **then**
    if *There exist two neighbours n_1 and n_2 of s_1 and s_2 respectively that are connected and are not common to s_1 and s_2* **then**
      Triangulate rectangular hole **(Figure 6d)**: Create two faces using s_1, s_2, n_1 and s_2, n_1, n_2;
    end
    else if *There exist two neighbours n_1 and n_2 of s_1 and s_2 respectively that are not connected between them and are not common to s_1 and s_2* **then**
      Triangulate pentagonal hole **(Figure 6e)**: Create three faces using s_1, s_2, n_2; s_1, n_2, n_3 and s_1, n_1, n_3;
    end
  end
else
  if *s_1, s_2 have two common neighbours n_1,n_2* **then**
    if *n_1 and n_2 are already connected **(Figure 6c)*** **then**
      Edge between n_1 and n_2 is removed;
      Remove coincident faces to n_1 and n_2;
      Create two faces using n_1, s_1, s_2 and n_2, s_1, s_2;
    else
      Create an edge between s_1 and s_2;
      Create two faces using n_1, s_1, s_2 and n_2, s_1, s_2;
      (Figure 6b);
    end
  else
    Create edge between s_1 and s_2;
    if *s_1 and s_2 have one common neighbour n_1 **(Figure 6a)*** **then**
      Create a face f using s_1,s_2 and n_1;
    end
  end
end

Algorithm 1: Pseudo-code of the extended CHL stage.

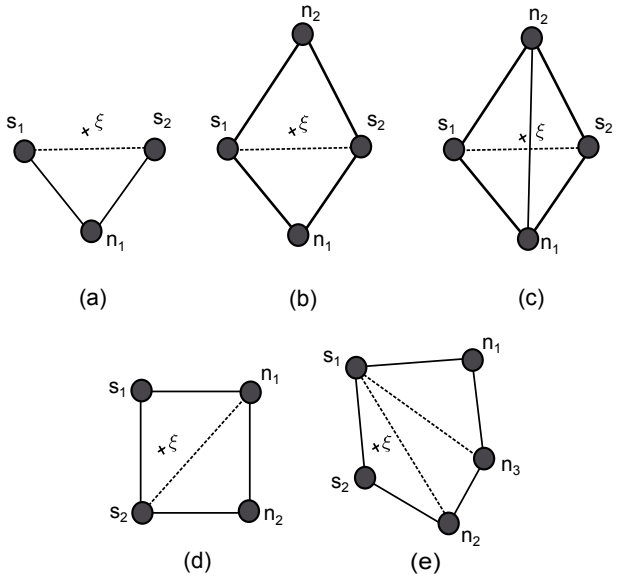

Fig. 6. Considered situations for edge and face creation during the extended CHL.

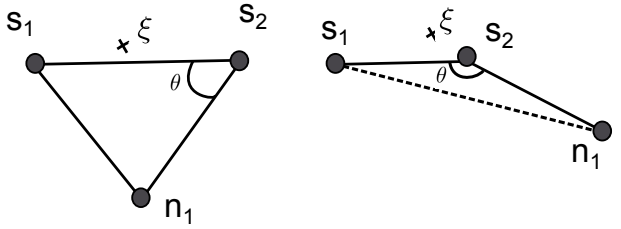

Fig. 7. Edge removal constraint based on the Tales sphere. Left: The triangle formed by these 3 neurons is close to a right triangle, therefore it is not removed. Right: The edge connecting s_1 and n_i is removed as the angle formed by vectors $s_2 - s_1$ and $n_i - s_1$ is larger than $3/4\pi$. Moreover, the triangle formed by these edges is also removed.

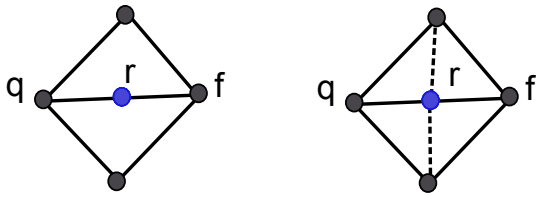

Fig. 8. Face creation process during the insertion of new neurons. Left: neuron insertion between the neuron q with highest error and its neighbour f with highest error. Right: four new triangles and two edges are created considering r, q and r.

B. Inserting and deleting neurons

The neuron insertion process was also modified. Every time a neuron is inserted in the network, an edge between the neurons with highest error is removed and therefore, triangles incident to this edge are also removed. If it is possible new faces are created along with the new neuron (Figure 8).

Finally, if the given number of neurons is reached, all the input patterns are presented to the network in order to close possible gaps and holes that were generated during the

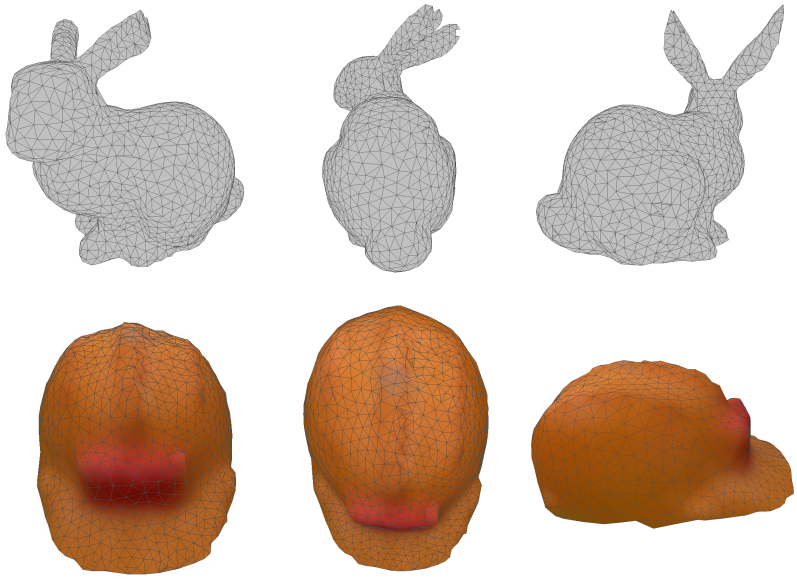

Fig. 9. Reconstructed models using our extended GNG method for face reconstruction and without applying post-processing steps. Top: Stanford bunny. Bottom: builder helmet.

learning process.

V. Experiments

In this section, different experiments are shown validating the capabilities of our extended GNG method to create 3D meshes. The proposed method is also able to learn colour information and therefore create coloured 3D meshes. 3D models were rendered using colour information stored in the neurons and a triangle smooth-shading technique. The method was tested using different point clouds. Some of these point clouds were obtained using the Kinect sensor (builder helmet and person). Other models as the dinosaur and the foot were acquired using other 3D sensors as the Minolta laser scanner and a foot digitizer. Finally, the well known Stanford bunny was also tested.

Figure 9 and 10 show the ability of the proposed method to create colour meshes of different types of models. It can be seen how most holes and gaps showed in Figure 5, generated by existing extensions of the GNG method for surface reconstruction, were not generated using the proposed method.

The proposed extension is also able to generate 3D meshes with different resolutions and therefore, detail level. Figure 11 shows the builder helmet model reconstructed using different number of neurons, creating meshes with different level of detail.

However, the proposed method still produced some small gaps in the generated 3D reconstructions. Figure 12 shows small gaps and holes created in some of the experiments carried out. These are caused by the randomness of the network learning stage. Moreover, in some cases triangles are removed caused by the edge removal ageing scheme, which also is responsible of the good level of adaptation and

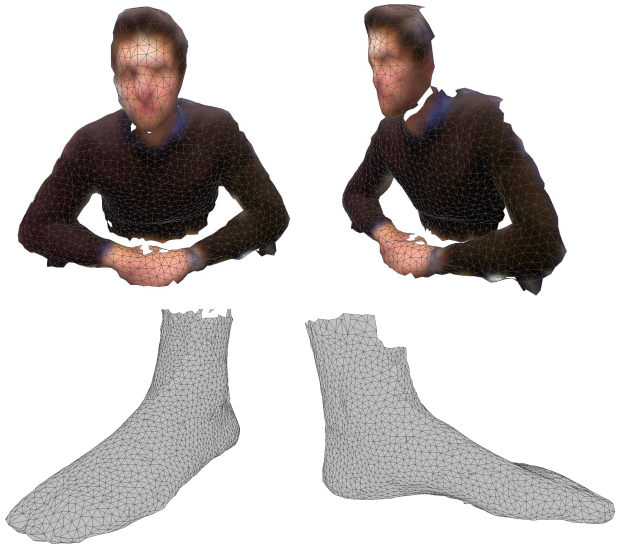

Fig. 10. Reconstruction of 3D human models with the proposed method. Top: 3D model of a person. Bottom: digitized human foot.

relationship between neurons. Despite this fact, the proposed method is valid for many computer vision applications. 3D triangular faces are used in many 3D descriptors which are often used in object and scene recognition applications.

VI. Conclusions and Future Work

In this paper we have presented a novel computing method to create complete meshes from unorganized raw noisy 3D data. Previous knowledge about the sensor is not necessary. It has been demonstrated how Growing Self-Organizing Maps

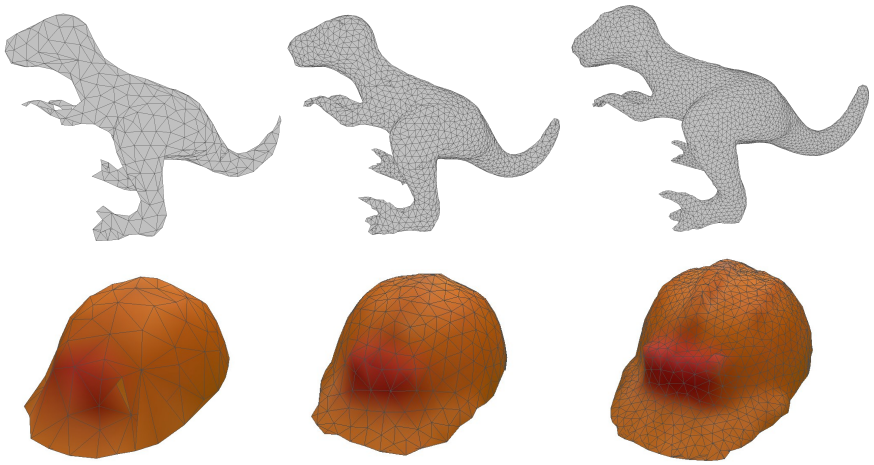

Fig. 11. Different 3D reconstructions of a the builder helmet model using various network sizes. Left: 3D reconstruction using 250 neurons and 200 input patterns. Middle: 3D reconstruction using 1000 neurons and 500 input patterns. Right: 3D reconstruction using 2500 neurons and 1000 input patterns.

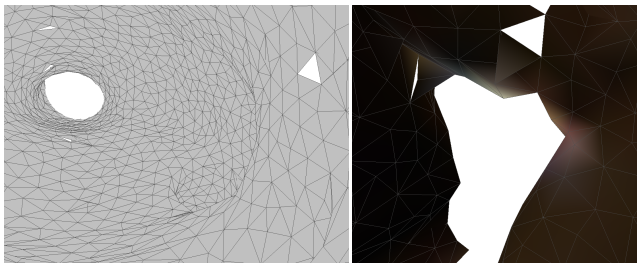

Fig. 12. Small gaps produced by our extended GNG method for 3D surface reconstruction.

(GSOM) are capable to represent noisy 3D data distributions. Neurons' codevector has also been modified adding colour components to their weights. Due to this modification, the algorithm is able to adapt its structure to the input space topology during the learning step and also to learn and store colour from the observation. This eliminates the necessity to add post-processing steps to add colour information to the final representation.

Furthermore, the GNG algorithm has been also extended considering the creation of triangular faces during the learning stage. In contrast with existing methods, our extension allowed to create complete triangular meshes with colour information during the learning stage, not requiring any post-processing steps to close gaps and holes. The method was validated with several models ranging from scanned objects to body parts like the foot.

Future work includes the adaptation and application of the proposed method for 3D scene reconstruction tasks.

Acknowledgements

This work was partially funded by the Spanish Government DPI2013-40534-R grant. Experiments were made possible with a generous donation of hardware from NVDIA.

References

[1] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Surface reconstruction from unorganized points," *SIGGRAPH Comput. Graph.*, vol. 26, no. 2, pp. 71–78, Jul. 1992. [Online]. Available: http://doi.acm.org/10.1145/142920.134011

[2] N. Amenta, S. Choi, and R. K. Kolluri, "The power crust," in *Proceedings of the sixth ACM symposium on Solid modeling and applications*, ser. SMA '01. New York, NY, USA: ACM, 2001, pp. 249–266. [Online]. Available: http://doi.acm.org/10.1145/376957.376986

[3] Y. Yu, "Surface reconstruction from unorganized points using self-organizing neural networks yizhou yu," in *In IEEE Visualization 99, Conference Proceedings*, 1999, pp. 61–64.

[4] A. Junior, A. D. D. Neto, and J. de Melo, "Surface reconstruction using neural networks and adaptive geometry meshes," in *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*, vol. 1, 2004, pp. –807.

[5] B. Fritzke, "Growing cell structures - a self-organizing network for unsupervised and supervised learning," *Neural Networks*, vol. 7, pp. 1441–1460, 1993.

[6] I. Ivrissimtzis, W.-K. Jeong, and H.-P. Seidel, "Using growing cell structures for surface reconstruction," in *Shape Modeling International*, May 2003, pp. 78 – 86.

[7] T. Martinetz and K. Schulten, "Topology representing networks," *Neural Networks*, vol. 7, no. 3, 1994.

[8] J. Barhak, "Freeform objects with arbitrary topology from multirange images." Ph.D. dissertation, Israel Institute of Technology, Haifa, Israel, 2002.

[9] B. Fritzke, *A Growing Neural Gas Network Learns Topologies*. MIT Press, 1995, vol. 7, pp. 625–632.

[10] A.-M. Cretu, E. M. Petriu, and P. Payeur, "Evaluation of growing neural gas networks for selective 3D scanning," in *Proc. Int. Workshop Robotic and Sensors Environments ROSE 2008*, 2008, pp. 108–113.

[11] Y. Holdstein and A. Fischer, "Three-dimensional surface reconstruction using meshing growing neural gas (MGNG)," *Vis. Comput.*, vol. 24, pp. 295–302, March 2008.

[12] R. L. M. E. Do Rego, A. F. R. Araujo, and F. B. De Lima Neto, "Growing self-reconstruction maps," *Trans. Neur. Netw.*, vol. 21, no. 2, pp. 211–223, Feb. 2010. [Online]. Available: http://dx.doi.org/10.1109/TNN.2009.2035312

[13] V. L. D. Mole and A. F. R. Araújo, "Growing self-organizing surface map: Learning a surface topology from a point cloud," *Neural Comput.*, vol. 22, no. 3, pp. 689–729, Mar. 2010. [Online]. Available: http://dx.doi.org/10.1162/neco.2009.08-08-842