2D Weather product visualization using Marching Squares algorithm

Igor Antolović[1], Dejan Rančić[2], Vladan Mihajlović[3], Dragan Mihić[4], Marija Đorđević[5]

Abstract – **In this paper we present a efficient way for 2d weather product visualization by using an approach which is based on the *Marching Squares* algorithm. We examine the possibilities of extracting both isolines and color filled isoareas as well as improvements in order to avoid the default broken polyline *Marching Squares* result and force more smooth visualization. By using different sampling resolutions and parameters we test and evaluate this approach on real weather temperature products.**

Keywords – Isoline, Marching squares, besier curves

I. INTRODUCTION

Modern climatology depends heavily on weather tracking GIS (Geographic Information System) [1] applications and their ability to process, analyze and visualize weather data. This kind of systems range from simple applications for processing of climate data to more complex weather forecasting systems [2]. The nature of weather data can be vector (e.g. wind) or scalar (temperature, pressure, precipitation etc.) thus emphasizing the need for various visualization algorithms aimed towards both 2D and 3D data as well as vector and scalar data types. HAISIS (Integrated Hail Suppresion Information System) [3,4,5] is one such system. It is developed at the Faculty of Electronic Engineering in Niš, at the Computer Graphic and Geographic Information Laboratory (CG&GIS Lab) to satisfy requirements of Republic Hidrometeorological Service of Serbia (RHMSS). It is operating more that 10 years and it supports LIC (Line Integral Convolution) [6] methods for vector data visualization, 2D isoline tracing, 3D isosurface extraction, 3D wind visualization and various other visualization algorithms offering climatology experts powerful tools for weather/climate tracking and analysis.

This paper focuses explicitly on visualization of scalar data using the *Marching Squares* algorithm considering it can be directly applied on 2D scalar weather product visualization. Although the most common version of this algorithm is the classic linear segment approach, we will further cover possible besier curve advancements in order to generate smooth isolines as well as the extended *Marching Squares* version suited for extraction of color filled isoareas.

Considering that the isoline extraction and visualization concept represents a scalar field visualization problem in the second part of this paper we give a short review of available techniques for scalar field visualization. The third part gives a detailed description of the Marching Squares algorithm followed by the fourth part which presents practical implementation results in combination with mean temperature test data. Finally the fifth part consists of conclusions and notes on future work.

II. 2D SCALAR FIELD VISUALIZATION CONCEPTS

A discrete 2D scalar field is essentially a (mxn) grid where every grid cell takes a value from a real set of numbers:

$$S:(x_0,..,x_m)\times(y_0,..,y_n)\rightarrow R \quad (1)$$

There are basically two types of 2D scalar data visualization techniques: color mapping and isoline extraction as shown on Fig.1.

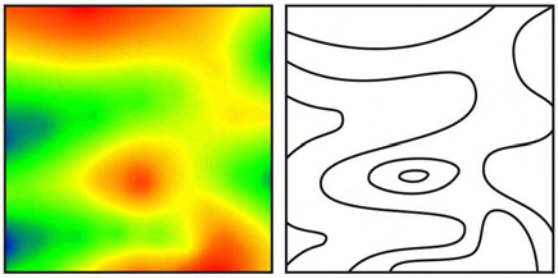

Fig.1. Color mapping (left) and isoline extraction (right)

Color mapping approach can be most easily implemented and uses a quite straightforward approach. The basic idea behind color mapping is to traverse through all scalar field values and map those values into colors. Additionally color values can be interpolated in order to get a smoother output. Usually instead of RGB (Red Green Blue) color model, HSB (Hue Saturation Brightness) is used considering that it defines more practical color scheme. Also an interesting

[1]Igor Antolović is with the Faculty of Electronic Engineering, Aleksandra Medvedeva 14, 18000 Nis, Serbia, E-mail: igor.antolovic@elfak.ni.ac.rs.

[2]Dejan Rančić is with the Faculty of Electronic Engineering, Aleksandra Medvedeva 14, 18000 Nis, Serbia, E-mail: dejan.rancic@elfak.ni.ac.rs.

[3]Vladan Mihajlović is with the Faculty of Electronic Engineering, Aleksandra Medvedeva 14, 18000 Nis, Serbia, E-mail: vladan.mihajlovic@elfak.ni.ac.rs.

[4]Dragan Mihić is with the Republic Hydrometeorological Service of Serbia and South East European Virtual Climate Change Center, E-mail: dragan.mihic@hidmet.gov.rs

[5]Marija Đorđević is with the Republic Hydrometeorological Service of Serbia and South East European Virtual Climate Change Center, E-mail: mapuja@gmail.com

recommendations on using colors with climate data can be found in [7].

On the other hand when isoline extraction is considered it is necessary to start from its very definition. For a given isovalue (c) an isoline consists of a set of points defined by:

$$I = \{(x, y) \mid S(x, y) = c\} \quad (2)$$

Meaning that isolines consists of all points that share a constant isovalue (c). On the other hand when it comes to practical isoline extraction there are two basic methods:

- Contour tracing
- Isoline segment extraction using *Divide and Concur approach*

Contour tracing is the most intuitive isoline extraction method and it is based on the idea of tracing values on a scalar field grid from a single seed point until a boundary is reached or a contour loop is detected. Recent work on this can be found in [8]. By using this method a continuous poly-line is produced which can be later additionally interpolated and smoothed. The common problem in this approach can be found in the process of seed point determination. Also in the case of multiple contour tracking, multiple track markers must be used. As previously stated, one practical application of this method can be found also in the HASIS 3DI system where this approach is used to extract dBz contours for given isolevels.

The *Divide and Concur* approach relies on the fact that the problem of isoline extraction can be simplified by extracting isoline segments in single scalar field cells and afterwards join those segments into continuous isolines [9]. This method is applicable for 3D as well 2D scalar fields. The 3D case of this method is well known by the name of *Marching Cubes* [10] and it is used for fast extraction of isosurfaces. In this paper we will further explore and describe it's simplified 2D version also known as the *Marching Squares* algorithm.

III. MARCHING SQUARES

The main idea behind the *Marching Squares* algorithm is to simplify the isoline extraction and visualization procedure by processing scalar field cells one at a time.

It can be easily concluded that the values in the cell corners can be above or below the given iso-value. Considering this fact a set of 16 possible cases can be isolated as shown on Fig. 2. Furthermore a simple scheme can be involved in order to index these cases where 0 means the corner value is below the required isovalue and 1 means the corner value is above the required isovalue.

This coding scheme can be uses to precalculate index tables in order to drastically speedup the process of identifying the current cell case.

The steps of processing the current cell can be described as follows:

- Compare the current cell corners against the required isovalue and determine the cell index.
- Use the calculated index in order to extract information of which edges are intersected by the isoline from the precalculated edge table.
- By using linear interpolation determine the exact points of intersection of the isoline and the cell edges.
- Connect the intersection points in order to get the current isoline segment.

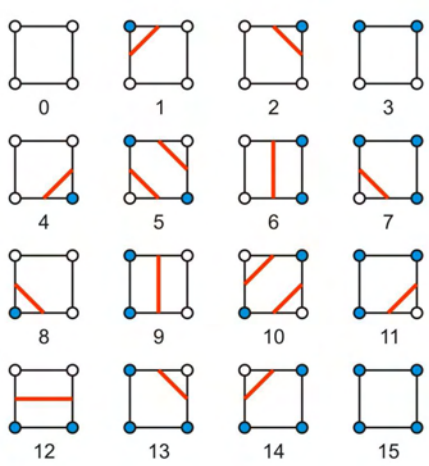

Fig.2. Marching Squares 16 possible cases

Upon processing all cells and drawing all isoline segments a full isoline set will be created as shown on Fig.3.

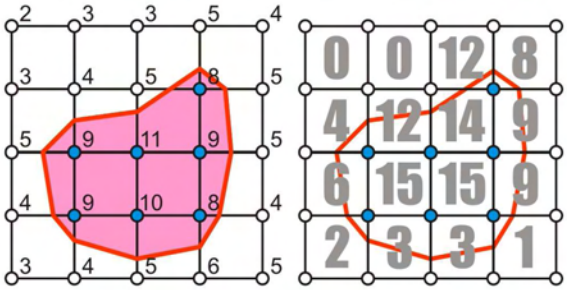

Fig.3. Example of isoline extraction sheme for isovalue (7)

This approach is very fast but it has two main disadvantages:

- The quality of the generated isolines depends on the resolution of the grid. Since the processing of a single cell produces line segments it is obvious that on small resolution grid isolines will be broken polylines.
- The second disadvantage is that there is no information about a continuous isoline.

It is obvious that for fixed edge intersection points there can be lot of solutions for the isoline segments but it is necessary to use the approach which provides the smoothest output.

On order to increase the isoline quality a common method is to take advantage of the gradient property of the scalar field. Basically the gradient is a vector which is pointed towards the greatest differential change in the given point of a scalar field.

Let the 2D scalar field be defined as:

$$S = f(x, y) \tag{3}$$

The gradient can be calculated as follows:

$$\nabla S = \vec{i}\frac{\partial S}{\partial x} + \vec{j}\frac{\partial S}{\partial y} \tag{4}$$

In the practical case where a discrete scalar field is used the gradient components can be calculated by using central differences as follows:

$$g(x, y)_x = \frac{S(x+1, y) - S(x-1, y)}{2\Delta x} \tag{5}$$

$$g(x, y)_y = \frac{S(x, y+1) - S(x, y-1)}{2\Delta y} \tag{6}$$

By using these equations we can easily calculate gradient vectors of the cell corners and furthermore calculate gradient vectors on isoline and grid intersection points by using linear interpolation. The most interesting property of the gradient vector is that it is perpendicular to the isoline passing through a given point, meaning that the vector normal to the gradient vector acts as a tangent vector of the isoline. This information can be used to construct smooth isoline segments instead of just using straight lines. In order to accomplish this, besier curves can be used considering the flexible relation between besier control points and besier curve normals.

The process of smooth isoline segment (Fig. 4) construction can be described in several steps as follows:

- Calculate gradient vectors in cell corner points,
- Calculate gradient vectors on isoline intersection points by interpolating between corner gradient vectors,
- Calculate normal vectors of previously calculated isoline gradient vectors,
- Find normal vector intersection M,
- Calculate besier control points as half distances between isoline intersection points and normal intersection point M.

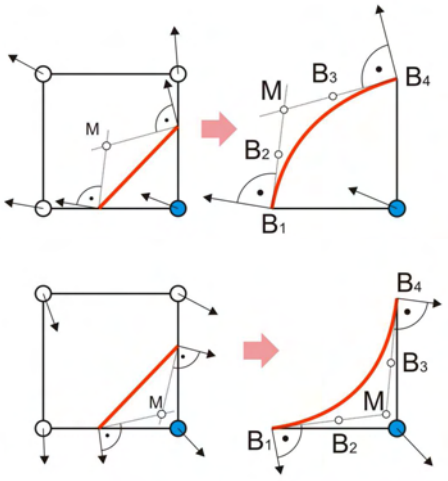

Fig.4. Besier curve construction examples

By processing all cells in this way and generating besier isoline segments, a drastic increase in quality can be noticed as the isolines show smooth flow through the cells.

Finally a simple polygonal scheme can be derived from the original *Marching Squares* set as shown on Fig.5. This new set can be used to generate color filled isoareas. By combining multiple layers of isoareas a multicolor isoarea can be generated as shown on Fig.6. The only problem with this approach is that it is order dependent. It is obvious that the isoareas must be sorted in increasing isovalues going from bottom to top.

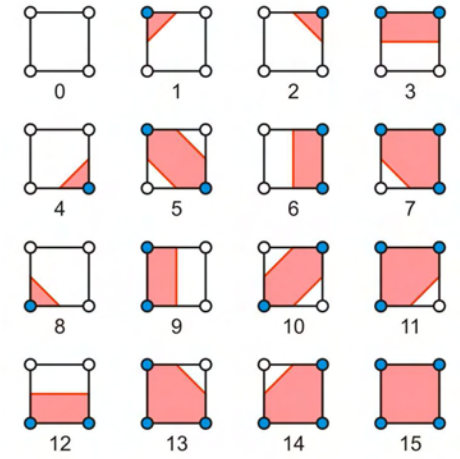

Fig.5. Polygonal Marching Squares scheme

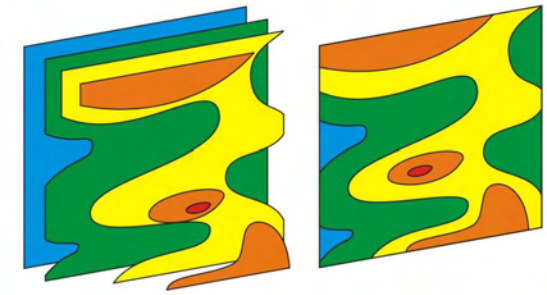

Fig.6. Multilayered polygonal isolevel approach

IV. RESULTS

For testing purposes as an input data a average temperature dataset was used which was sampled on a 64x64 grid. In order to visually compare the quality of the generated isolines both classic linear and advanced besier approach was used. The results are shown on Fig. 7. What can easily be noticed is that for higher sampling resolutions as well small cell sizes the difference between the advanced smoothing and classic linear approach becomes indistinguishable. On the other hand for higher zoom levels the broken polyline appearance of the classic linear method becomes more obvious as shown on Fig. 7.c). Beside the classic isoline extraction, the multilayered isoarea approach was also implemented so depending on the need either of this visualization techniques can be used. For example in lots of cases a combination of isolines and terrain map is can be used as shown on Fig.7.b)

a) b) c)

Fig.7. a) Smooth isoareas with normals b) Non-smooth isolines on map c) Comparison of smooth and non-smooth details

V. Conclusion

Efficient visualization of climate data products is a basic requirement in all weather tracking and analysis GIS applications. Taking into account the nature of this data, two groups of climate variables can be distinguished. The first group consists of scalar climate variables (temperature, pressure, precipitation etc.) and the second group consists of vector climate variables (e.g. wind).

In this paper we considered visualization techniques for the first group of variables. This kind of visualization has already been identified as a scalar field visualization problem which is most commonly solved by using isoline extraction approach. Among two basic isoline extraction methods (by contour tracing and by using the *Marching Squares* divide and concur method) we choose the second method considering that it relies on precalculated tables thus providing great speed as well as multiple isoline extraction.

As a proof of concept we implemented the Marching Squares Algorithm and additionally considered isoline segment smoothing by using besier curves. Finally we compared the results of classic linear *Marching Squares* approach with besier curve based results. As it is shown the classic linear approach suffers from a broken polyline look compared to the smooth besier based approach. Furthermore we compared isoareas and isoline visual results as well variants of isoline on map backgrounds. Upon evaluating this results we can conclude that the *Marching Squares* algorithm give a rather elegant, fast, and visually satisfying solution to the isoline extraction problem. What is also important to emphasize is that the final quality of the generated isolines depends directly no the sampling grid resolution thus the future work on this matter will include considerations of techniques for isoline extraction on lower sampling grid resolutions.

Acknowledgement

This paper was realized as a part of the project "Studying climate change and its influence on the environment: impacts, adaptation and mitigation" (43007) financed by the Ministry of Education and Science of the Republic of Serbia within the framework of integrated and interdisciplinary research for the period 2011-2014.

References

[1] D. Rančić, A. Dimitrijević, V. Mihajlović, "GIS and Virtual Reality Systems Integration", ICEST 2004, Bitola, Macedonia, pp. 313-316, 2004.

[2] L. A. Treinish, “Visual data fusion for applications of high-resolution numerical weather prediction”, *IEEE Proceedings of the conference on Visualization '00*, 2000, pp. 477 - 480.

[3] V. Mihajlović, S. Đorđević-Kajan, D. Rančić, B. Predić, I. Antolović, P. Eferica, Z. Babić, “Architecture of HASIS-3D System Designed for Hail Suppression Purposes”, Proceedings of ICEST 07, Ohrid, jun 2007.

[4] Đorđević-Kajan, S., Milovanović, B., Rančić, D., Kostić, A., Stanić, Z., Stoimenov, L., “Hail Suppression Information System of Radar Center”, GIS/LIS'96, Budapest Hungarz, June 1996, pp.102-111.

[5] Rančić, D., Smiljanić, M., Đorđević-Kajan, S., Kostić, A., Eferica, P., Vuković, P., Vučinić, Z., “Radar Data Processing for Cloud Seeding in Hail Suppression Information System”, RADME 98

[6] M. Kovačević, V. Mihajlović, I. Antolović, D. Rančić, Z. Babić: LIC based Visuelization of Air Flow in Clouds, Conference on Computer Science and Information Technologies – YUINFO 2010, Kopaonik, Serbia, 3 – 6 March 2010. (In Serbian).

[7] American Meteorological Society, Guidelines for using Color to Depict Meteorological Information: IIPS Subcommittee for Color Guidelines. 1993

[8] Fu Chang , Chun-Jen Chen , Chi-Jen Lu, A linear-time component-labeling algorithm using contour tracing technique, Computer Vision and Image Understanding, v.93 n.2, p.206-220, February 2004

[9] G. Cottafava, G. Le Moli, "Automatic Contour Plotting", *Comm. ACM 12*, 7 (July 1969), pp. 386-391.

[10] Lorensen W, Cline H. Marching cubes: a high resolution 3D surface construction algorithm. Computer Graphics 1987; 21(4):163–9