Particle Tracing in σ-Transformed Grids using Tetrahedral 6-Decomposition

I. Ari Sadarjoen[1], Alex J. de Boer[1,2], Frits H. Post[1], and Arthur E. Mynett[2]

[1] Dept. of Computer Science, Delft University of Technology
PO Box 356, 2600 AJ Delft, The Netherlands
email: {I.A.Sadarjoen, A.J.deBoer, F.H.Post}@cs.tudelft.nl
[2] WL | Delft Hydraulics, Strategic R&D
PO Box 177, 2600 MH Delft, The Netherlands
email: Arthur.Mynett@wldelft.nl

Abstract. *Particle tracing in curvilinear grids often employs decomposition of hexahedral cells into 5 tetrahedra. This method has shortcomings when applied to σ-transformed grids, a grid type having strongly sheared cells, commonly used in hydrodynamic simulations. This paper describes an improved decomposition method into 6 tetrahedra. It is shown that this method is robust in σ-transformed grids, however large the shearing. Results are presented of applying the technique to a real world simulation. Comparisons are made between the accuracy and speed of the 5-decomposition and the 6-decomposition methods.*

1 Introduction

Particle tracing is an important technique for visualization of velocity vector fields resulting from computational fluid dynamics (CFD) simulations [3]. The basic particle tracing technique is based on a stepwise numerical integration of the velocity field. Especially the numerical integration techniques have been well studied [7, 8]. A source of complications is the use of irregular grids in CFD simulations, such as structured curvilinear grids. The cells of such grids are hexahedra with arbitrarily deformed geometry; the adjacency structure (topology) of the grid cells is regular. But the irregular geometry causes critical operations in particle tracing algorithms to be more complex, such as finding which cell contains a given point.

Several solutions have been proposed for this problem. One is to transform the deformed hexahedral cells to cubes (computational space or C-space), perform the path integration in a regular grid, and transform the results back to the original deformed grid. Another solution is to perform the tracing directly in the deformed grid (physical space or P-space), using a decomposition of the hexahedral cells into tetrahedra. We have analysed and compared these techniques in an earlier paper [6], and found that the P-space approach gave the best results both in accuracy and efficiency, if visualization is applied as a postprocessing activity following the actual computations, and C-space velocities are not directly available.

The P-space algorithms using a tetrahedral decomposition of hexahedral cells have gained acceptance [4], but some problems can occur in strongly deformed grid cells. In large-scale hydrodynamic simulations, the x- and y-dimensions are typically 2 to 3 orders of magnitude larger than the z-dimension. Such simulations often use so-called σ-transformed curvilinear grids, in which the hexahedral cells can be very flat and strongly deformed. This can cause the tetrahedral 5-decomposition to produce locally invalid results, and the P-space tracing algorithm to fail.

In this paper, we will examine the reasons for this failure in more detail, and propose a new decomposition which can be proven to maintain validity of the grid even for very strong deformations of σ-transformed grid cells. Section 2 gives some fundamentals of P-space particle tracing, and briefly surveys the 5-tetrahedron decomposition of hexahedral cells. After explaining the σ-transformation in Section 3, we analyse the problems occurring with the 5-decomposition. Section 4 presents the improved 6-decomposition, and experimental results are shown in Section 5. Conclusions and directions for further work are given in Section 6.

2 Fundamentals of Particle Tracing

The computation of a particle path is based on the integration of the ordinary differential equation

$$\frac{d\mathbf{x}}{dt} = \mathbf{v}(\mathbf{x}) \tag{1}$$

where $\mathbf{x}$ denotes the position of the particle, t is time, and $\mathbf{v}(\mathbf{x})$ the velocity field. The starting position $\mathbf{x}_0$ of the particle provides the initial condition at initial time t_0: $\mathbf{x}(t_0) = \mathbf{x}_0$. Subsequent points are calculated as

$$\mathbf{x}(t_{n+1}) = \mathbf{x}(t_n) + \int_{t_n}^{t_{n+1}} \mathbf{v}(\mathbf{x})dt \tag{2}$$

using a numerical integration method. The solution is a sequence of particle positions $(\mathbf{x}(t_0), \mathbf{x}(t_1), \ldots)$ at time steps $t_0, t_1, \ldots$

Particle Tracing in Regular Rectangular Grids

Let us first recall the structure of a particle tracing algorithm for regular rectangular (uniform) grids:

```
find cell containing initial position          (point location)
while particle in grid
    determine velocity at current position     (interpolation)
    calculate new position                     (integration)
    find cell containing new position          (point location)
endwhile
```

Point location is the process of determining which cell contains a specified point. In uniform grids, this is easily accomplished by splitting the coordinates of a point into the integer cell indices (i, j, k) and fractional offsets (α, β, γ). *Interpolation* is the process of determining a data value at an arbitrary position in a given cell, using the surrounding grid nodes and the fractional offsets. In uniform grids, this is typically done with first-order trilinear interpolation.

Particle Tracing in Curvilinear Grids

In practice, many CFD applications do not use uniform grids, but *structured curvilinear grids*, consisting of deformed, hexahedral cells. An advantage of curvilinear grids is that they can follow the shape of curved or complex geometries such as airplane wings and coast lines. The disadvantage is that algorithms working on these grids are more complex, because the cells are no longer regular cubes, but they may be sheared and have curved faces.

One strategy often applied in many CFD simulation systems, is to transform the curvilinear grids in physical space to a uniform grid in a new domain, called *computational space*. Unfortunately, for visualization algorithms, this method did not turn out to be beneficial, as was investigated in detail in [6].

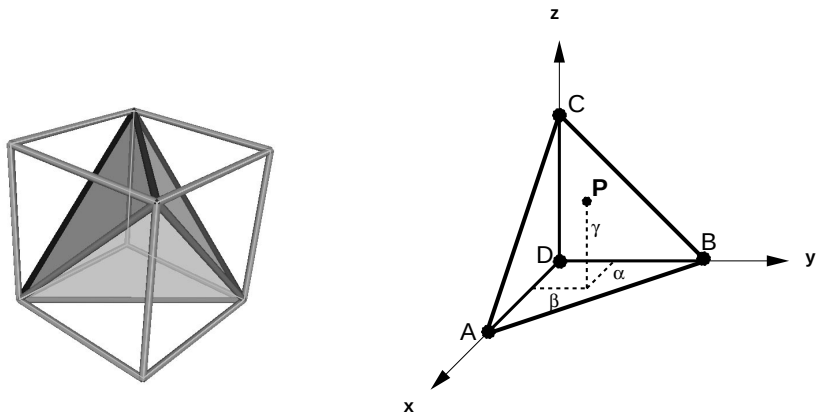
Another strategy is to calculate the particle path directly in physical space. This would avoid transformations between the two domains, although at the expense of more difficult point location and interpolation. Interpolation in curvilinear grids is more difficult, because the offsets are harder to determine in a curved cell. Point location in curvilinear grids is more difficult, because there is no longer a direct relation between the global coordinates of a point and the cell indices. Instead, a search must be performed, by checking for several cells if they contain the point. Usually, there is a previous position in a known cell, which is connected to the new position in the unknown cell by a line. Along this line, the algorithm traverses subsequent adjacent cells by intersecting the line with the cell faces and checking which adjacent cell has that face in common.

Decomposition into 5 Tetrahedra

One way to cope with curved cells works by decomposing the hexahedral cells into tetrahedra. The advantages of tetrahedra is that they are convex and planar, which facilitates containment tests and face intersection tests.

The simplest and most efficient scheme is to decompose the hexahedral cells into 5 tetrahedra, henceforth called the *5-decomposition*. Figure 1a shows a cube which is decomposed into 1 (shaded) center tetrahedron and 4 corner tetrahedra. In a structured grid, the decomposition can be done in two directions. To ensure connection of cell faces and to avoid overlapping cells, these two directions should be alternated in adjacent cells, as shown in Figure 1b.

In tetrahedra, interpolation and point location are performed as follows. *Interpolation* in tetrahedra is done using linear interpolation. Figure 1c shows a tetrahedron ABCD, where α, β, γ denote the fractional offsets in the tetrahedron, with the restriction that $\alpha + \beta + \gamma \leq 1$. If $\mathbf{v}_A$ is the data value in node A, $\mathbf{v}_B$



(a) 1 center tetrahedron and 4 corner tetrahedra

(b) Linear interpolation in tetrahedra

Fig. 1. Decomposition of a hexahedral cell into 5 tetrahedra

the data value in node B, etc., then the interpolated value $\mathbf{v}_P$ in some position P in the tetrahedron is: $\mathbf{v}_P = \mathbf{v}_D + \alpha(\mathbf{v}_A - \mathbf{v}_D) + \beta(\mathbf{v}_B - \mathbf{v}_D) + \gamma(\mathbf{v}_C - \mathbf{v}_D)$. The fractional offsets (α, β, γ) may be found by inverting the interpolation of the position of P in the tetrahedron.

$$P = D + \alpha(A - D) + \beta(B - D)\gamma(C - D) \tag{3}$$

$$(\alpha, \beta, \gamma) = (A - D|B - D|C - D)^{-1}(P - D) \tag{4}$$

Point location is again done by traversing cells from a previous position, although not entire cells are traversed, but the tetrahedra into which they are decomposed.

3 σ-Transformed Grids

Point location using tetrahedral 5-decomposition regularly fails, especially in a specific type of grids known as *σ-transformed grids*. In our test cases, up to 40%(!) of the particles were caught in an infinite loop between two cells, or stopped completely. Before explaining the cause of these problems, let us first describe this type of grids.

σ-Transformed grids are frequently used in hydrodynamic simulations of shallow waters, such as marine coasts or estuaries. They consist of stacked 2D xy-layers, each of which is a well-formed quadrangular mesh with curved and usually orthogonal grid lines. Corresponding nodes in different layers have identical x,y coordinates. In the vertical direction, the grid lines are straight and parallel to the z-axis. σ-coordinates are defined relative to the local water elevation ζ and depth

d, as $\sigma = \frac{z-\zeta}{\zeta+d}$. The top grid layer, where $\sigma = 0$, follows the free water surface, which usually only varies gradually. The bottom layer, where $\sigma = -1$, follows the sea floor geometry, which typically has strongly varying depths throughout the model. The layers in between are constructed with a prescribed distribution. Figure 2 shows one possible distribution of 6 layers. Figure 3 shows a sea floor geometry and a vertical grid slice in the Lith harbour data set, which was used in a simulation and visualization project at WL | Delft Hydraulics [5] (see Figure 9 for colour).

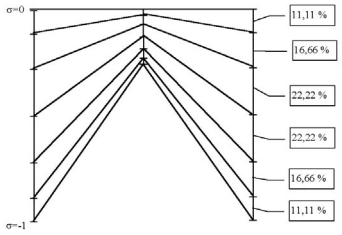


Fig. 2. Distribution of a σ-transformed grid with 6 layers

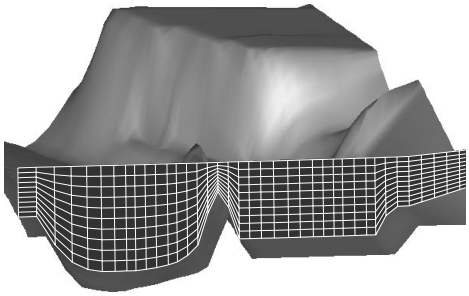


Fig. 3. σ-transformed grid, with a sea floor geometry and a vertical grid slice

In σ-transformed grids, many cells are sheared in the vertical direction, because the number of layers is constant, while the local depth varies, so parallel vertical edges often lie at very different depths. The shearing is increased as the cells are typically very thin in these applications: the model may be hundreds of kilometers wide and only tens of meters deep. Strongly sheared cells have some typical characteristics. In a normal cell, the orientation of the center tetrahedron is as shown in Figure 4a, but in a strongly sheared cell, the center tetrahedron has been turned inside out, as shown in Figure 4b. The top faces BEG and DEG now lie at the bottom, while the bottom faces BDE and BDG lie on top. The edges BD and GE have crossed each other. This is possible because the center tetrahedron has edges spanning the entire cell.

These strongly sheared cells result in two problems with the 5-decomposition. One problem is that the center tetrahedron overlaps with corner tetrahedra, and even with a neighbouring cell. As a consequence, the point location algorithm cannot determine a unique tetrahedron which contains a given point, and exits. The second problem is that particles may get caught in an infinite loop between two tetrahedra. Due to the reversed orientation of the center tetrahedron, the point location algorithm fails to find the correct outgoing face, and therefore the correct adjacent tetrahedron. As a consequence, the algorithm moves from a corner tetrahedron to the center tetrahedron, and then return to the corner tetrahedron where it came from, instead of proceeding to the next one. In this way, it will continue moving back and forth forever.

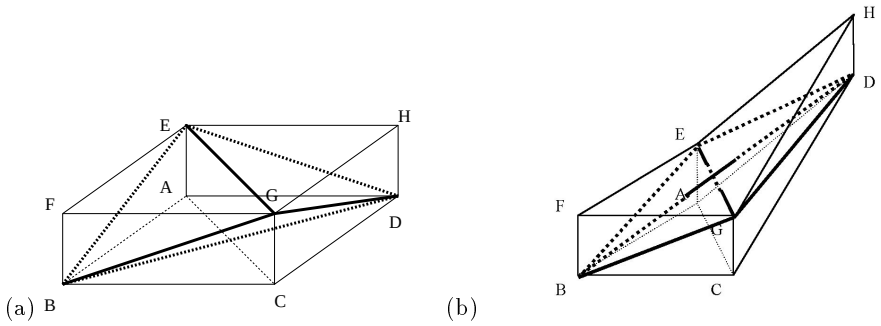


Fig. 4. (a) Normal (a) and (b) reversed tetrahedral orientation in 5-decomposed hexahedra

The frequency at which these problems occurred, varied between 4% and 40% of the particles, depending on the data set and particle source locations. In some cells, the problems might be solved by changing the decomposition direction, since the problem is direction dependent. But then the problem would probably occur in other cells, because the direction is chosen globally for the entire grid.

4 Tetrahedral 6-Decomposition

An apparent solution to the point location problems comes to mind: scale or shear a deformed cell such that the twisted orientation of the edges and tetrahedra is avoided. However, scaling or shearing grid cells amounts to applying a computational space algorithm: the grid is transformed to a different domain, where the cells are regular and rectangular. We chose not to do this in Section 2 because of the loss of accuracy and efficiency [6].

A better approach is to use a different tetrahedral decomposition. A systematic overview of the possibilities can be found in [1]. A hexahedron can be decomposed into 5, 6, and any even number between 12 and 24 tetrahedra. For reasons of efficiency and storage space, the preferred approach is the decomposition into 6 tetrahedra, henceforth called the *6-decomposition*. Figure 5 shows how this is accomplished: a hexahedral cell is decomposed into two three-sided prisms, each of which is decomposed into 3 tetrahedra. Just like the 5-decomposition, the 6-decomposition has two directions: each face diagonal can be chosen in two ways. But an advantage of the 6-decomposition method is that it does not require the directions to alternate for adjacent cells.

The main advantage of this 6-decomposition method is that it solves the point location problems. There is no longer a center tetrahedron whose edges span the entire cell, and which may cross each other when the cell is sheared in the vertical direction. Figure 6 shows the 6-decomposition in a normal cell, and in a sheared cell comparable to Figure 4b. It can be clearly seen that the tetrahedra in the

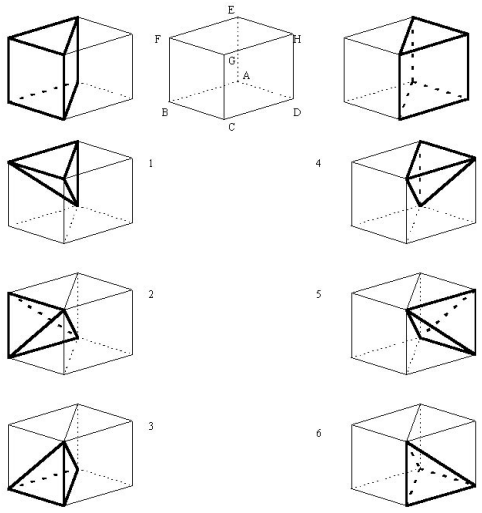


Fig. 5. Decomposition of a hexahedron into 6 tetrahedra

sheared prism retain their orientations, since the hatched planes AGH and ADG retain their orientations and relative positions. No tetrahedron has been turned inside out. It can be shown that this method is robust: the tetrahedra will never change orientation, no matter how large the shearing is, as long as the edges are only displaced in the vertical direction (as is the case with σ-transformed grids).

5 Results

The technique described above was implemented in a set of AVS/Express modules called `PLANKTON-97` [2]. Modules were developed for interactively placing point, line, and plane particle sources, for calculating the particle paths, and for creating animations. To evaluate the technique, three types of tests were performed: a functional test and a speed test.

Functional Test

To test the system, we have performed tests in artificial and real world data sets. Whereas the 5-decomposition method would fail in 4% to 40% of the released particles, the 6-decomposition method did not have any problems in tracing the particles through strongly deformed cells. Here, we show an example of particles traced in the Lith Harbour data set. The grid is a σ-transformed curvilinear grid with 121x40x10 cells. At the grid nodes, velocity and turbulence intensity were defined. In this data set, 100 particles were released in a horizontal plane. Figure 7 shows the particles rendered as arrows. It is clearly visible that

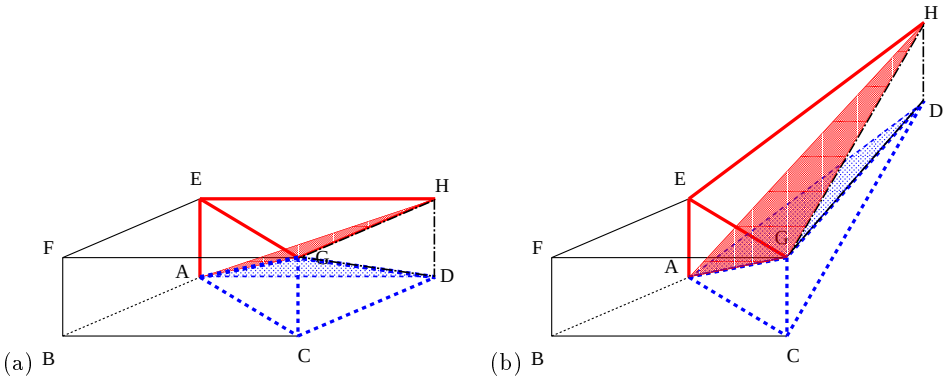


Fig. 6. Tetrahedral orientation in 6-decomposed (a) normal cell and (b) sheared cell

the 5-decomposition method failed to trace 14 out of 100 particles, rendered as circles. The 6-decomposition was successful for all particles (not shown).

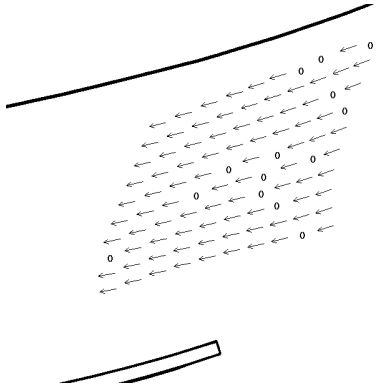


Fig. 7. The 5-decomposition method failed to trace 14% of the particles in Lith Harbour

Figure 8 shows another example where the 6-decomposition was successful (see Figure 10 for colour). Here, instead of the particles themselves, the traced particle paths were rendered, coloured with velocity magnitude. In addition, a bounding box and a sea floor geometry were rendered to increase the sense of depth.

Speed Test

To compare the speed of both algorithms, we measured the execution times necessary for creating 100 animation frames. In the Lith Harbour data set, particles were released from one source located near the center of the data set.

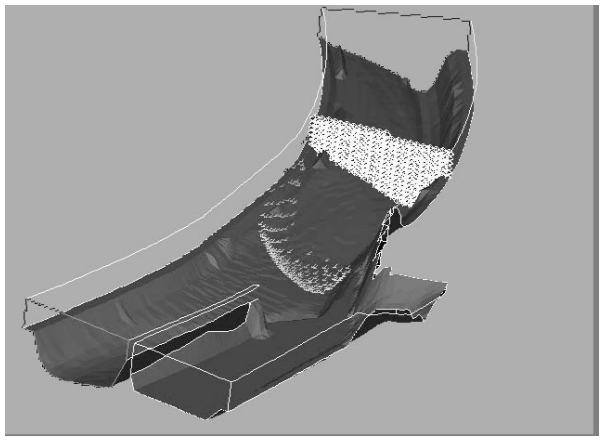


Fig. 8. Particles successfully traced with the 6-decomposition method

After every 25 frames, a new particle was released. For each frame, we calculated 25 integration steps of $\Delta t = 10s$, which amounts to 2500 integration steps. The machine used was an SGI Indigo^2 with a MIPS R10000 processor at 195 MHz. Note that the 6-decomposition is slightly faster than the 5-decomposition, even though it creates more tetrahedra. This is due to a simple optimization for avoiding redundant operations in traversing the decomposed cell. Table 1 lists the test results, which were performed several times and averaged, to obtain accurate measurements.

	5-decomposition	6-decomposition
execution time (s)	10.61	10.47
# traversed cells	85	85
# traversed tetrahedra	234	243

Table 1. Speed comparison of 5-decomposition and 6-decomposition methods

6 Conclusions and Future Work

It has been shown that decomposition of hexahedral cells in σ-transformed grids into 6 tetrahedra is better than decomposition into 5 tetrahedra. Particles whose paths could not be traced due to the limitations of the 5-decomposition, could be successfully traced with the 6-decomposition method. The 6-decomposition method has shown to be robust, regardless of the amount of vertical shearing of the cells.

In practice, the grids used in hydrodynamic simulations have more application-specific features, such as missing (dry) grid points, thin dams, boundary points requiring special care, etc. However, these do not fall within the scope of this paper, but some solutions are presented in Chapter 6 of [2].

The tetrahedral decomposition can be used in unstructured tetrahedral grids with only slight modifications, if face/cell adjacency information is available for traversing the grid.

Acknowledgements

The work described here is a part of the second author's Master's thesis work [2], performed at WL | Delft Hydraulics. We wish to thank Irving Elshoff of WL | Delft Hydraulics for his supervision and valuable suggestions.

References

1. G. Albertelli and R.A. Crawfis. Efficient subdivision of finite-element datasets into consistent tetrahedra. In R. Yagel and H. Hagen, editors, *Proc. Visualization '97*, pages 213–219. IEEE Computer Society Press, 1997.
2. A.J. de Boer. Reconstructie en uitbreiding van Plankton in AVS/Express. Master's thesis, Delft University of Technology, January 1998. In Dutch.
3. A.J.S. Hin and F.H. Post. Visualization of turbulent flow with particles. In G.M. Nielson and R.D. Bergeron, editors, *Proceedings Visualization '93*, pages 46–52. IEEE Computer Society Press, Los Alamitos, CA, 1993.
4. D.N. Kenwright and D.A. Lane. Optimization of time-dependent particle tracing using tetrahedral decomposition. In G.M. Nielson and D. Silver, editors, *Proc. Visualization '95*, pages 321–328. IEEE Computer Society Press, 1995.
5. D.G. Meijer. Lock approach second ship lock at Lith. Scale model investigation and DELFT3D-calculcations. Technical report, WL | Delft Hydraulics, June 1995. In Dutch.
6. A. Sadarjoen, T. van Walsum, A.J.S. Hin, and F.H. Post. Particle tracing algorithms for 3D curvilinear grids. In *Proc. 5th EuroGraphics Workshop on Visualization in Scientific Computing*, 1994.
7. T. Strid, A. Rizzi, and J. Oppelstrup. Development and use of some flow visualization algorithms. In *Computer Graphics and Flow Visualization in Computational Fluid Dynamics*, Lecture Series 1989-07. Von Kármán Institute for Fluid Dynamics, 1989.
8. C. Teitzel, R. Grosso, and T. Ertl. Efficient and reliable integration methods for particle tracing in unsteady flows on discrete meshes. In W. Lefer and M. Grave, editors, *Visualization in Scientific Computing '97*, pages 31–41. Eurographics, Springer, 1997.

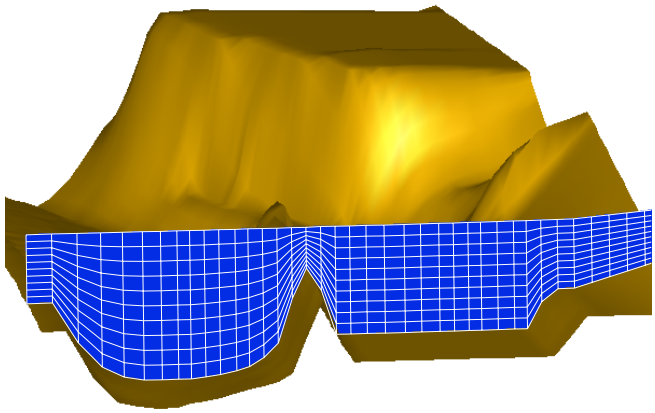


Fig. 9. Sigma-transformed grid in Lith harbour

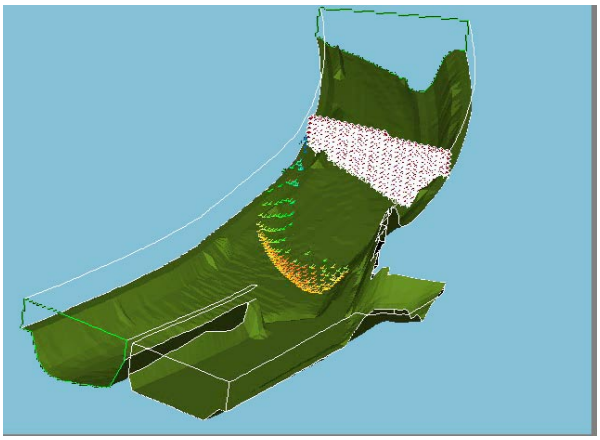


Fig. 10. Particles successfully traced with the 6-decomposition method