Advanced Structural Mechanics 1 (2023) 102-114

Advanced Structural Mechanics

journal homepage: https://asm.sku.ac.ir/

A space-time formulation for the level-set evolution simulation

Mohsen Lashkarbolok[a]*

[a]*Department of Civil Engineering, Faculty of Engineering, Golestan University, Aliabad Katoul 4941147994, Iran*

Article received:2022/11/10, Article revised: 2023/01/27, Article accepted: 2023/02/09

ABSTRACT

Level-set method is a flexible and efficient tool for structural topology optimization. Here, a technique based on the space-time formulation in the context of an integral-free meshfree method in a Eulerian approach is presented for interface tracking by the use of the level-set method. We use the radial basis functions for function approximation in the space-time domain. The technique does not need to march in time to obtain interface development. It is obtained instantly at all defined times by solving sparse and symmetric linear equations. In this paper, the time-slab concept is applied to reduce the computational cost of the procedure. The proposed technique provides the possibility of applying effective adaptive refinement techniques in domain discretization for the transient problem. A vectorized formulation of the proposed approach is illustrated in this paper making it efficient in calculation and readable for programming. Two problems with analytical solutions are solved to evaluate the accuracy of the presented technique.

Keywords: Interface tracking; Level-set method; Free-surface modeling; Moving boundaries

1. Introduction

The application of the level-set method for structural topology optimization has been introduced and developed by decent studies in [1-3]. A comprehensive review of the technique is provided in [4]. The result of the topology optimization technique demonstrates the outstanding flexibility of handling topological changes, fidelity of boundary representation and degree of automation [2]. The level-set method, itself, has been introduced by Osher and Sethian [5]. It has been used broadly for interface tracking [6-9]. In this method the interface evolution in time is obtained by the numerical solution of an advection equation for a signed-distance function. In the traditional level-set methods, the level-set function may lose its property during evolution in time and we need to reshape or re-initialize it [10]. The re-initialization procedure may become complicated, expensive, and ensue subtle side effects [10]. Li et al.

* Corresponding author at: Department of Civil Engineering, Faculty of Engineering, Golestan University, Aliabad Katoul, Iran.
E-mail address: m.lbolok@gu.ac.ir
DOI: 10.22034/asm.2023.14141.1007: https://asm.sku.ac.ir/article_11385.html

presented a variational formulation for geometric active contours that forces the level-set function to be close to a signed-distance function, and eliminates the need for re-initialization procedure [10]. They controlled deviation of the level-set function from signed-distance function using a penalizing parameter. This method was improved later by Li et al. [11]. Zhang et al. presented a re-initialization-free level-set evolution by adding a diffusion term in the original level-set advection equation. The amount of smoothing diffusion term was controlled by a parameter [12]. Wang et al. reviewed some recent developments on re-initialization elimination methods and proposed an enhanced distance regularized level-set evolution technique [13]. In the present paper we use a space-time formulation for the numerical solution of the level-set function advection equation. Space-time formulation has been introduced in [14-16] and applied in many scientific problems, mostly in the context of the Finite Element Method (FEM). Hulbert and Hughes used a space-time finite element method to solve elastodynamics problems that include sharp gradients due to propagating waves [17]. They proved unconditional stability and high-order of accuracy of the space-time finite element formulation. Tezduyar et al. presented a finite element strategy for free-surface flows based on the deforming spatial-domain/space-time procedure [18, 19]. Behr used the same technique for three-dimensional finite element computation of an unsteady, incompressible free-surface flow [20]. Tezduyar et al. extended this method for computation of fluid-structure interaction problems [21]. Some recent works in the field of space-time finite element formulation are found in [22-26]. Zwart et. al. developed an integrated space-time finite volume method for fluid flow simulation [27]. Klaij et. al. presented a space-time discontinuous Galerkin finite element method for the solution of compressible Navier-Stokes equation [28]. Netuzhylov and Zilian developed a space‐time meshfree collocation method for solving systems of non‐linear ordinary and partial differential equations by a consistent discretization in both space and time [29]. Dohr et al. introduced a new parallel solver for the weakly singular space–time boundary integral equation for the heat equation [30]. Zhang et al. proposed new space–time spectral and structured spectral element methods for high order problems. They introduced a family of new basis functions that leads to a diagonal system for the fourth order problems with constant coefficients [31]. Reinstädler et al. applied the four-dimensional space–time finite elements for fluid–structure interactions together with the single-phase level-set method to investigate the dynamics of three-dimensional landslides interacting with flexible walls [32]. In the present paper, we use the radial point interpolation method based on the radial basis functions for function approximation in both time and space. The least-squares technique is applied to find those solutions minimizing the sum of squared residuals coming from interpolation procedure. The least-squares techniques have been used broadly in the finite element method for the solution of PDEs (for instance [33-35]). The same technique has been applied to minimize the squared residuals coming from moving least squares interpolation method [36, 37] and radial point interpolation methods [38, 39]. In the present paper we discretize the space-time domain using the distribution of nodes. This point-wise approach is desirable when you extend the method for three-dimensional problems. Since we consider time as an independent dimension the same as space dimensions, solving a two-dimensional problem requires a three-dimensional domain discretization. Although the proposed formulation produces a sparse and symmetric linear system of equations, the computational costs will increase tremendously in a fine discretization. In order to make the presented approach computationally less expensive, without losing accuracy, we applied a time-slab idea [40]. Results at all times in the domain problem, or at all times in a time-slab, are found simultaneously. A detailed vectorized formulation of the proposed approach is presented. It is shown that most of the computational procedure comprises some algebraic operation on highly sparse matrices.

Nomenclature

u	velocity components in x direction
v	velocity components in y direction
φ	signed distance function
m	number of neighbor nodes
N	number of nodes in domain discretization
ψ	shape function matrix
β	penalty coefficient
Δ	uniform discretization size of a slab

2. Governing equation

Consider a two-dimensional domain, here in the x-y plane, which is discretized using some nodes as shown in Fig.1. Initial position of interface is shown in the figure. Signed-distance function for i^{th} node, φ_i , is defined as the shortest distance between the node and nearest node in the interface. We take the signed-distance function negative for nodes in the interface boundaries. It means, $\varphi_i = d_i$ and $\varphi_j = -d_j$. Assume the interface moves with a defined velocity field (u, v) at all nodes in the domain. u and v are velocity components in x and y directions, respectively. The interface evolution in time is obtained by solving the following partial differential equation:

$$\frac{\partial \varphi}{\partial t} + u\frac{\partial \varphi}{\partial x} + v\frac{\partial \varphi}{\partial y} = 0 \tag{1}.$$

The initial condition, in the *x-y* plane, or boundary condition in the *x-y-t* space is written as:

$$\varphi = \varphi_b \text{ at } t=0 \tag{2}.$$

Here, $\varphi = 0$ contour-line at each time shows the position of interface at that time.

3. Solution strategy

Here we treat the time dimension the same as space dimension, x or y. By this way, a two-dimensional problem in the *x-y* plane turns into a three-dimensional problem in the *x-y-t* space and the initial condition in the *x-y* plane for φ becomes a boundary condition in the *x-y-t* domain. Fig. 2 shows the three-dimensional space-time domain of a two-dimensional plane problem. We discretize the computational space-time domain using the scatter nodes as shown in Fig. 3.

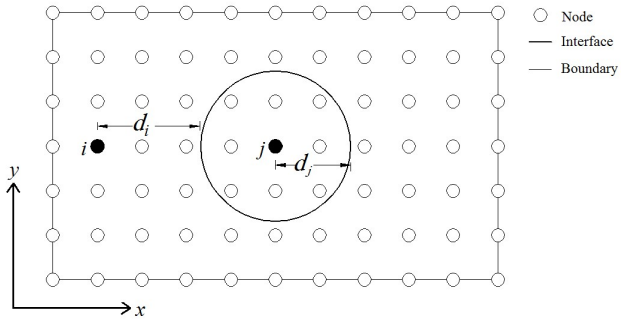

Fig. 1. Initial interface position in a domain

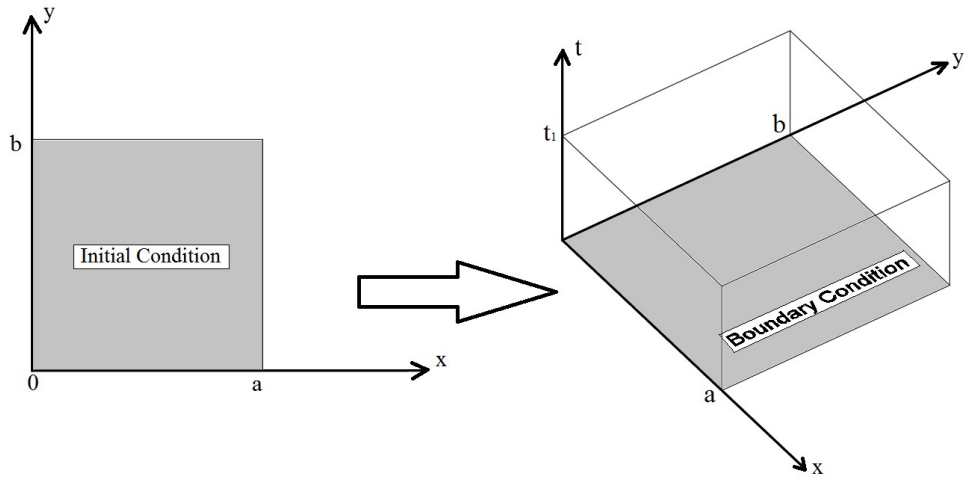

Fig. 2. Three-dimensional time-space domain of a two-dimensional space problem

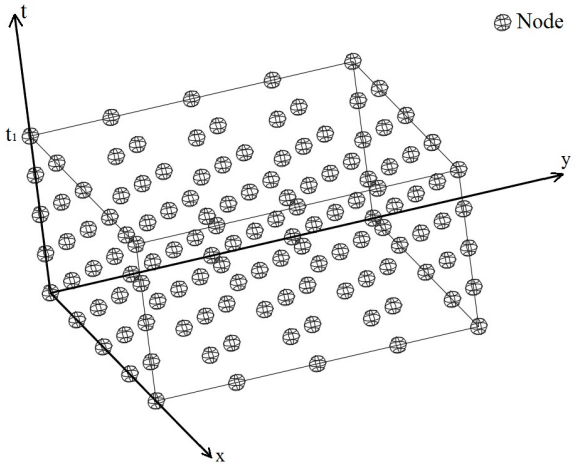

Fig. 3. Discretization of a domain in the x-y-t space using the scatter nodes

The value of the unknown function, φ, at i^{th} node depends on the value of φ at its neighbours node. Assume m neighbour nodes affect the value of φ_i. The value of φ_i is interpolated as:

$$\varphi_i = \alpha_0 + \alpha_1 x_i + \alpha_2 y_i + \alpha_3 t_i + \alpha_4 (x_i)^2 + \alpha_5 (y_i)^2 + \alpha_6 (t_i)^2 + \alpha_7 x_i y_i t_i + \sum_{k=1}^{m} \lambda_k \Phi(\frac{\ell_{ik}}{2r_i}) \quad (3).$$

where, r_i is the radius of the influence domain of i^{th} node as shown in Fig.4. ℓ_{ik} is the Euclidean distance between the i^{th} node and its k^{th} neighbour:

$$\ell_{ik} = \sqrt{(x_k - x_i)^2 + (y_k - y_i)^2 + (t_k - t_i)^2} \quad (4).$$

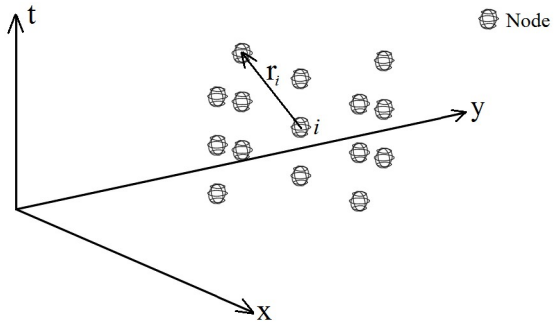

Fig. 4. Neighbours and radius of influence domain of i^{th} node, r_i

The function Φ is a Radial Basis Function (RBF). There are different RBFs used for curve fitting [41]. Here, we use a quartic function for Φ ,[41], as:

$$\Phi(d) = \frac{2}{3} + \frac{9}{2}d^2 + \frac{19}{3}d^3 - \frac{5}{2}d^4; \quad 0 \le d \le 1 \tag{5}.$$

Polynomial terms in the interpolation function, Eq.(3), ensure the reproduction of the parabolic field (C^2 consistency). It is reported, as we concluded, that the addition of polynomials can also improve the accuracy of the results and reduce the sensitivity of the shape parameters in RBFs to the solution accuracy [41]. The unknown coefficients $\alpha_0, \alpha_1, ..., \alpha_7$ and $\lambda_1, \lambda_2, ..., \lambda_m$ are obtained by solving the following set of equations [41] :

$$\varphi_i = \alpha_0 + \alpha_1 x_i + \alpha_2 y_i + \alpha_3 t_i + \alpha_4 (x_i)^2 + \alpha_5 (y_i)^2 + \alpha_6 (t_i)^2 + \alpha_7 x_i y_i t_i + \sum_{k=1}^{m} \lambda_k \Phi(\frac{\ell_{ik}}{2r_i}), i = 1, ..., m \tag{6}.$$

We need eight more equations to obtain the unknown coefficients. They are presented as [41]:

$$\sum_{i=1}^{m} \lambda_i = 0 \;,\; \sum_{i=1}^{m} \lambda_i x_i = 0 \;,\; \sum_{i=1}^{m} \lambda_i (x_i)^2 = 0 \;,\; \sum_{i=1}^{m} \lambda_i y_i = 0 \;,\; \sum_{i=1}^{m} \lambda_i (y_i)^2 = 0 \;,\; \sum_{i=1}^{m} \lambda_i t_i = 0 \;,\; \sum_{i=1}^{m} \lambda_i (t_i)^2 = 0 \;,$$
$$\sum_{i=1}^{m} \lambda_i x_i \, y_i \, t_i = 0 \tag{7}.$$

Now we have $m+8$ equations to obtain $m+8$ unknown coefficients. We substitute the unknown coefficient into Eq. (3) to derive the following relationship:

$$\varphi_i = \sum_{k=1}^{m} \psi_k^i \, \varphi_k \tag{8}.$$

where, ψ is a function of (x, y, t) and is called shape function. ψ_k^i is the value of ψ at k^{th} neighbour of the i^{th} node. More detailed formulation is provided in [41]. Here we put the values of shape functions at all points into a matrix as:

$$\boxed{\psi} = \begin{bmatrix} \psi_{1,1} & \psi_{1,2} & \cdots & \psi_{1,N} \\ \psi_{2,1} & \psi_{2,2} & \cdots & \psi_{2,N} \\ \vdots & \vdots & \cdots & \vdots \\ \psi_{N,1} & \psi_{N,2} & \cdots & \psi_{N,N} \end{bmatrix} \tag{9}$$

where, N is the number of nodes which are used for domain discretization. $\psi_{i,j}$ is the value of the i^{th} shape function at j^{th} node. In the same way, we can find derivatives of shape function matrices, $\boxed{\frac{\partial \psi}{\partial x}}$, $\boxed{\frac{\partial \psi}{\partial y}}$ and $\boxed{\frac{\partial \psi}{\partial t}}$. Since we use the small influence domain for each node, matrices are sparse. In this paper we choose *m=64* and *N=121203*, so the number of non-zero elements will be 7756992[†]. It means that, here, 99.95% of shape function matrices are empty. There are some efficient techniques for working with the mainly empty matrices or sparse matrices in the literature, e.g. [42]. Using the shape function matrices, we can write the following approximations globally as:

$$\frac{\partial \boldsymbol{\varphi}}{\partial t} = \boxed{\frac{\partial \psi}{\partial t}} \boldsymbol{\varphi} \tag{10}$$

$$\frac{\partial \boldsymbol{\varphi}}{\partial x} = \boxed{\frac{\partial \psi}{\partial x}} \boldsymbol{\varphi} \tag{11}$$

$$\frac{\partial \boldsymbol{\varphi}}{\partial y} = \boxed{\frac{\partial \psi}{\partial y}} \boldsymbol{\varphi} \tag{12}$$

where, $\boldsymbol{\varphi}$ is a vector which contains the value of φ at all nodes. We put approximations (10), (11) and (12) into Eq. (1) and Eq. (2) to find residual vectors, $\mathbf{R_1}$ and $\mathbf{R_2}$, as:

$$\boxed{\frac{\partial \psi}{\partial t}} \boldsymbol{\varphi} + u \boxed{\frac{\partial \psi}{\partial x}} \boldsymbol{\varphi} + v \boxed{\frac{\partial \psi}{\partial y}} \boldsymbol{\varphi} = \mathbf{R}_1 \tag{13}$$

$$\boxed{\psi_b} \boldsymbol{\varphi} - \boldsymbol{\varphi}_b = \mathbf{R}_2 \tag{14}$$

$\boxed{\psi_b}$ is a shape function matrix for boundary nodes. To build it, we substitute those elements of the matrix $\boxed{\psi}$ which are not associated with nodes at the boundaries by zero. The function J, which is the sum of the squared residuals, is written as:

$$J = \frac{1}{2}(\mathbf{R}_1^{\mathrm{T}} \mathbf{R}_1 + \beta \mathbf{R}_2^{\mathrm{T}} \mathbf{R}_2) \tag{15}$$

where, T is the transpose sign. In this study, the residuals in the boundaries are scaled by a very big number (β),

[†] At each row we have 64 non-zero elements. So, the number of non-zero elements in the shape function matrix is $64 \times 121203 = 7756992$

named penalty coefficient, to enforce the boundary conditions‡ [43]. J is calculated as:

$$J = \frac{1}{2}(\boldsymbol{\varphi}^{\mathrm{T}}\boxed{\frac{\partial \psi}{\partial t}}^{\mathrm{T}} + u\,\boldsymbol{\varphi}^{\mathrm{T}}\boxed{\frac{\partial \psi}{\partial x}}^{\mathrm{T}} + v\,\boldsymbol{\varphi}^{\mathrm{T}}\boxed{\frac{\partial \psi}{\partial y}}^{\mathrm{T}})(\boxed{\frac{\partial \psi}{\partial t}}\boldsymbol{\varphi} + u\boxed{\frac{\partial \psi}{\partial x}}\boldsymbol{\varphi} + v\boxed{\frac{\partial \psi}{\partial y}}\boldsymbol{\varphi}) + \ldots$$
$$\frac{1}{2}\beta(\boldsymbol{\varphi}^{\mathrm{T}}\boxed{\psi_b}^{\mathrm{T}} - \boldsymbol{\varphi}_b{}^{\mathrm{T}})(\boxed{\psi_b}\boldsymbol{\varphi} - \boldsymbol{\varphi}_b) \tag{16}.$$

where, $\boldsymbol{\varphi}$ which minimizes the function J, is the desired solution. It is found as:

$$\frac{\partial J}{\partial \boldsymbol{\varphi}} = 0 \tag{17}.$$

Or:

$$(\boxed{\frac{\partial \psi}{\partial t}}^{\mathrm{T}} + u\boxed{\frac{\partial \psi}{\partial x}}^{\mathrm{T}} + v\boxed{\frac{\partial \psi}{\partial y}}^{\mathrm{T}})(\boxed{\frac{\partial \psi}{\partial t}}\boldsymbol{\varphi} + u\boxed{\frac{\partial \psi}{\partial x}}\boldsymbol{\varphi} + v\boxed{\frac{\partial \psi}{\partial y}}\boldsymbol{\varphi}) + \beta\boxed{\psi_b}^{\mathrm{T}}(\boxed{\psi_b}\boldsymbol{\varphi} - \boldsymbol{\varphi}_b) \quad = 0 \tag{18}.$$

We can write the system of equations in a matrix form as:

$$\boxed{k}\,\boldsymbol{\varphi} = \mathbf{F} \tag{19}.$$

where:

$$\boxed{k} = (\boxed{\frac{\partial \psi}{\partial t}}^{\mathrm{T}} + u\boxed{\frac{\partial \psi}{\partial x}}^{\mathrm{T}} + v\boxed{\frac{\partial \psi}{\partial y}}^{\mathrm{T}})(\boxed{\frac{\partial \psi}{\partial t}} + u\boxed{\frac{\partial \psi}{\partial x}} + v\boxed{\frac{\partial \psi}{\partial y}}) + \beta\boxed{\psi_b}^{\mathrm{T}}\boxed{\psi_b} \tag{20}.$$

$$\mathbf{F} = \beta\boxed{\psi_b}^{\mathrm{T}}\boldsymbol{\varphi}_b \tag{21}.$$

The $\boxed{k}$ is sparse as all shape function matrices are sparse, and is symmetric as it comes from a least squares approach.

4. Time-slabs

In the presented approach, we consider time as an independent variable the same as space variables. Usually the space dimension of a problem has a limited value but the time dimension may be very long. Considering that the full length of time increases computational cost significantly, we slice the entire computational domain using some planes perpendicular to the t-axis and produce time-slabs. The result of the lower time-slab is a boundary condition for the upper time-slab. Time slabs for a sample space-time domain are shown in Fig. 5. The thickness of the time-slabs should be as small as possible to reduce the computational costs. In this paper, we first discretize the x-y plane by considering a fixed distance between nodes, Δ. We call this two-dimensional domain a layer. To produce a time-slab, we repeat each layer, n times with uniform distance of Δ in t direction (so every time-slab contains $n+1$

‡ This is the same as the well-known penalty method in the finite element method.

layers).

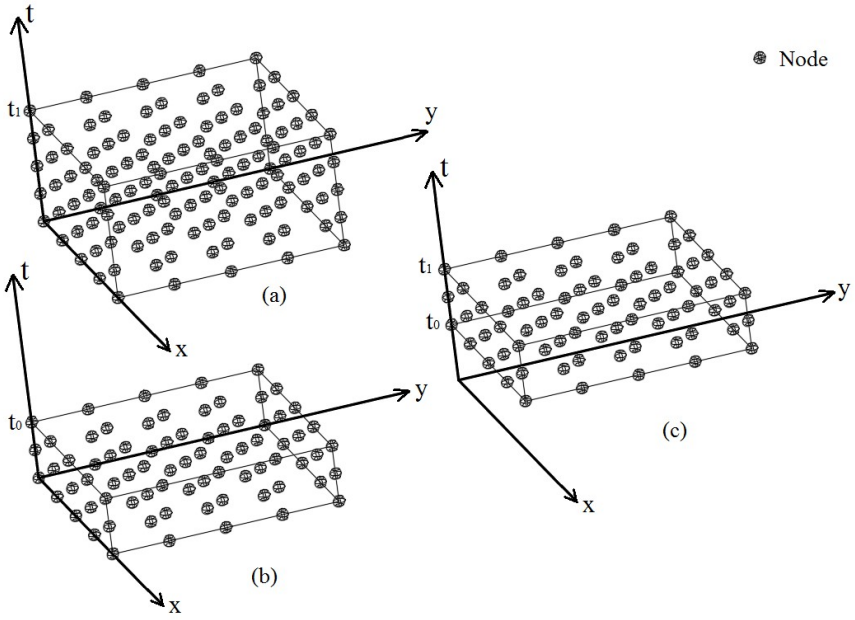

Fig. 5. Original domain discretization (a) and produced time-slabs, *n=2*, (b) and (c)

5. Numerical example

Here, two problems with analytical solutions are dealt with to show the performance of the proposed technique. We used the uniform discretization for simplicity, but since the formulation is based on a mesh and integral-free method, dealing with non-uniform nodal distribution is straightforward. The penalty coefficient, β, must be big enough to accurately enforce the boundary conditions. By choosing $\beta = 10^6$ boundary conditions satisfied for both proposed problems, larger values of β produce the same results. But much larger numbers (say 10^{20}) "deteriorate" the condition number of the coefficient matrices.

5.1. Problem 1.

Consider a circular interface in a rectangular domain, as shown in Fig. 6. The velocity field is constant in time and space domain by $u = 1, v = 0.2$. The centre of the initial circular interface is located at (0.5,0.5). The calculated new position of the interface at t=1 is shown in Fig. 7. using three different uniform sizes of discretization, $\Delta = 0.1, 0.05, 0.025$, along with the exact solution. The convergence of the numerical result to the exact solution by decreasing the discretization size is illustrated in Fig. 7. The proposed technique gives a reasonable result even at a coarse discretization.

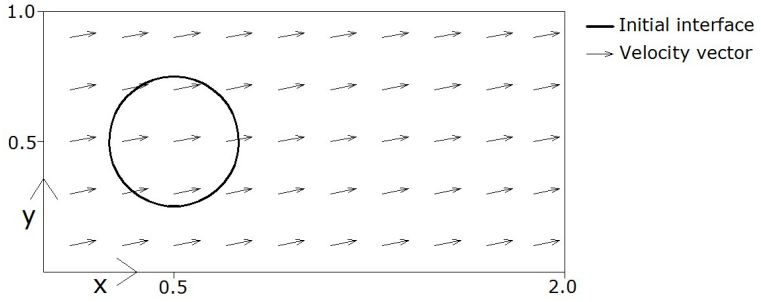

Fig. 6. Velocity field and initial interface location for problem 1.

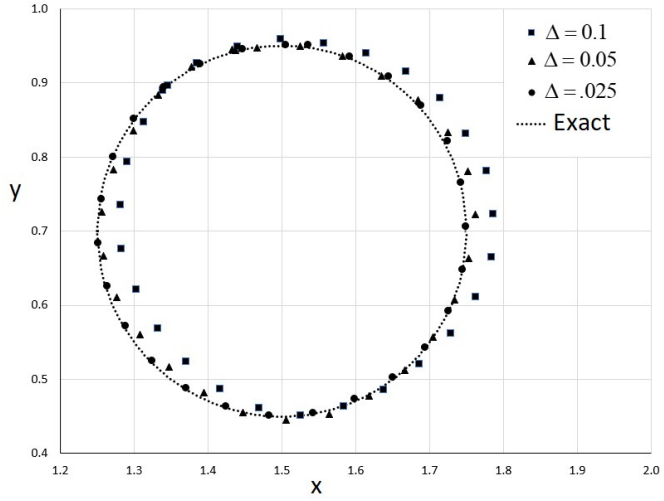

Fig. 7. The interface at t=1 using different discretization sizes along with the exact solution

5.2. Problem 2.

The presented method is evaluated by solving a single vortex flow problem [44]–[46]. In this problem the stretching of a circle with 0.3 diameter where placed at (0.5, 0.75) on a unit square under the following velocity field is simulated.

$$u(x,y) = -\sin^2(\pi x)\sin(2\pi y)\cos(\pi t / T)$$

$$v(x,y) = \sin(2\pi x)\sin^2(\pi y)\cos(\pi t / T)$$

where, T is the time that the flow returns to its initial state. Here we choose $T = 8$, indicating that the maximum stretching occurs at $t = 4$. The time-slab thickness is chosen to be 0.01 (or $\Delta = 0.005$). The resulting nodal distribution in a time-slab is shown in Fig. 8. According to our numerical experiences, the thicker time-slabs do not improve results significantly and increase the computational cost.

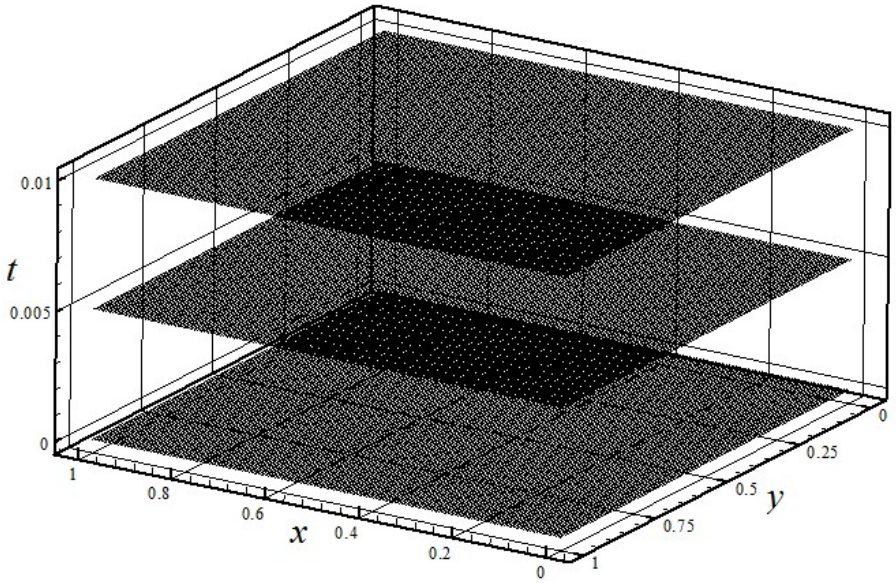

Fig. 8. A time-slab discretization uniformly distributes nodes, $\Delta = 0.005$, n=2

The stretching process of the initial circle at *t=2*, *4*, *6* and *8* is shown in Fig. 9. The shape of the final circle, the result obtained at *t=8*, well preserved the shape of the initial of the circle. Results have a good agreement with the results reported in [44-46].

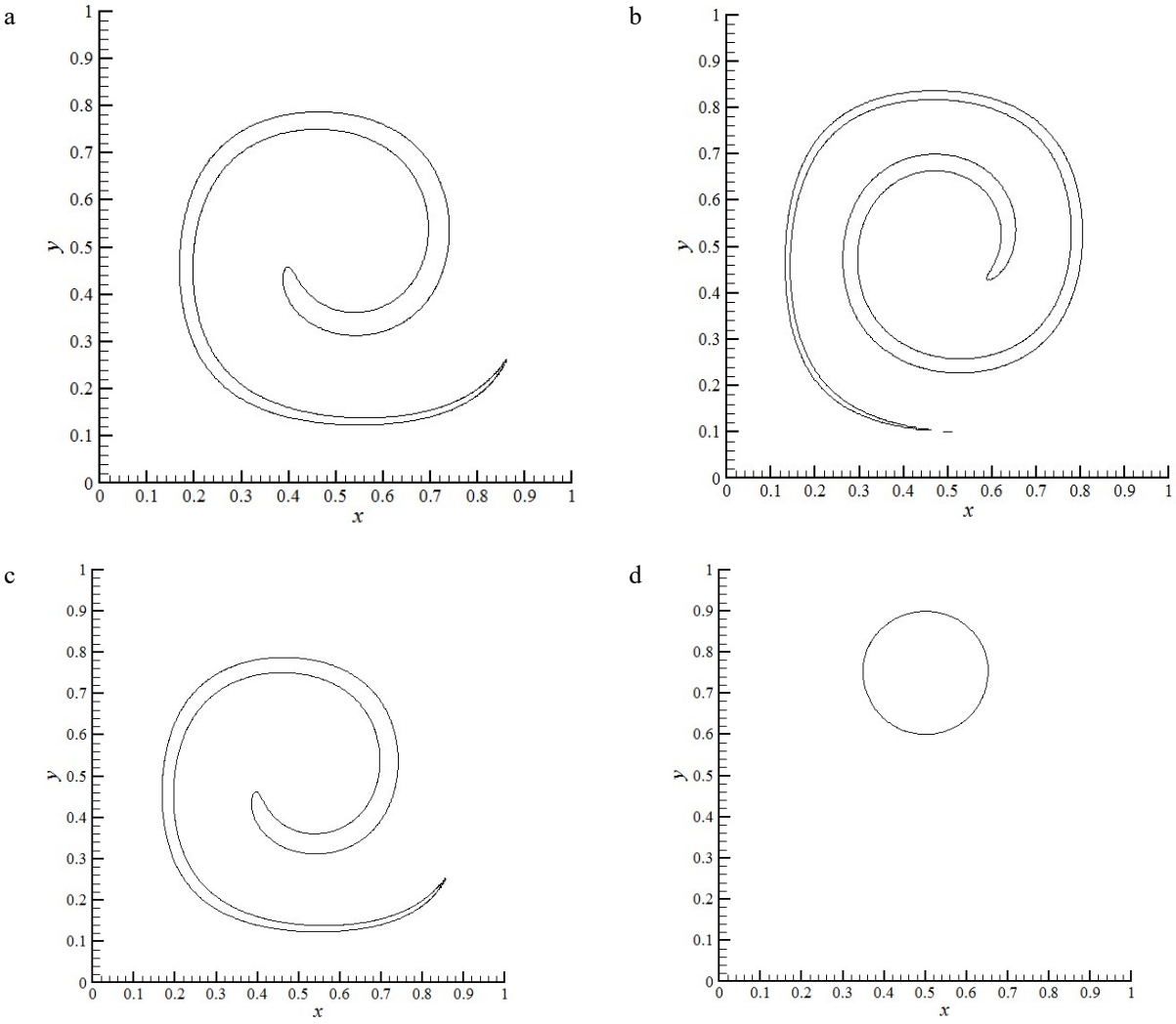

Fig. 9. (a) Zero contour-line ($\varphi = 0$) at *t=2*; (b) Zero contour-line at *t=4*; (c) Zero contour-line at *t=6;* (d) Zero contour-line at *t=8*

6. Conclusion

Level-set method is an efficient tool for structural topology optimization. In the presented paper, we introduce the time-space technique for the interface evaluation equation of the level-set method, which is the main novelty of the presented study. We do not use time marching. The results are obtained at the top of the time-slabs. A point-wise integral-free numerical method is employed based on the least-squares technique and radial basis functions. The idea of time-slabs is described in the paper and applied to reduce computational costs. The proposed approach produces symmetric and highly sparse matrices, making calculations efficient. Two problems are solved, and the accuracy of the method is evaluated. The results are promising, and further research is required to evaluate the possible benefits of the presented technique in structural topology optimization.

References

[1] Sethian, J.A.,Wiegmann, A., 2000. Structural Boundary Design via Level Set and Immersed Interface Methods. Journal of Computational Physics. 163(2), 489–528.

[2] Wang, M.Y., Wang, X., Guo, D., 2003. A level set method for structural topology optimization. Computer Methods in Applied Mechanics and Engineering. 192(1–2), 227–246.
[3] Allaire, G., Jouve, F., Toader, A.M., 2004. Structural optimization using sensitivity analysis and a level-set method. Journal of Computational Physics. 194(1), 363–393.
[4] van Dijk, N.P., 2013. Level-set methods for structural topology optimization: a review. Structural and Multidisciplinary Optimization. 48(3), 437–472.
[5] Osher, S., Sethian, J.A., 1988. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. Journal of Computational Physics. 79(1), 12–49.
[6] Osher, S., Tsai, R., 2003. Level Set Methods and Their Applications in Image Science. Communications in Mathematical Sciences. 1(4), 623-656.
[7] Li, C., Xu, C., Fox, M. D., 2010. Distance regularized level set evolution and its application to image segmentation. IEEE Transactions on Image Processing. 19(12), 3243-3254.
[8] van Dijk, N.P., Maute, K., Langelaar, M., Keulen, F., 2013. Level-set methods for structural topology optimization: a review. Structural and Multidisciplinary Optimization. 48(3), 437–472.
[9] Gibou, F., Fedkiw, R. and Osher, S., 2018. A review of level-set methods and some recent applications. Journal of Computational Physics. 353, 82–109.
[10] Li, C., Xu, C., Gui, C., Fox, M.D., 2005. Level set evolution without re-initialization: A new variational formulation. in Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR.
[11] Li, C., Xu, C., Gui, C., Fox, M.D., 2010. Distance regularized level set evolution and its application to image segmentation. *IEEE* Transactions on Image Processing. 19(12), 3243-3254.
[12] Zhang, L., Song, H., Zhang, D., 2013. Reinitialization-Free Level Set Evolution via Reaction Diffusion. IEEE Transactions on Image Processing. 22(1), 258–271.
[13] Wang, X., Shan, J., Niu, Y., Tan, L., Zhang, S.X., 2014. Enhanced distance regularization for re-initialization free level set evolution with application to image segmentation. Neurocomputing. 141, pp. 223–235.
[14] Argyris, J.H., Scharpf, D.W., 1969. Finite Elements in Time and Space. The Aeronautical Journal. 73(708), 1041–1044.
[15] Fried, I., 1969. Finite-element analysis of time-dependent phenomena. AIAA Journal. 7(6), 1170–1173.
[16] Oden, J.T., 1969. A general theory of finite elements. II. Applications, International Journal for Numerical Methods in Engineering. 1(3), 247–259.
[17] Hulbert, G.M., Hughes, T.J.R., 1990. Space-time finite element methods for second-order hyperbolic equations. Computer Methods in Applied Mechanics and Engineering. 84(3), 327–348.
[18] Tezduyar, T.E., Behr, M., Liou, J., 1992. A new strategy for finite element computations involving moving boundaries and interfaces—The deforming-spatial-domain/space-time procedure: I. The concept and the preliminary numerical tests', Computer Methods in Applied Mechanics and Engineering. 94(3), 339–351.
[19] Tezduyar, T.E., Behr, M., Liou, J., 1992. A new strategy for finite element computations involving moving boundaries and interfaces—The deforming-spatial-domain/space-time procedure: II. Computation of free-surface flows, two-liquid flows, and flows with drifting cylinders. Computer Methods in Applied Mechanics and Engineering. 94(3), 353–371.
[20] Behr, M., 2001. Stabilized space-time finite element formulations for free-surface flows. Communications in Numerical Methods in Engineering. 17(11), 813–819.
[21] Tezduyar, T. E., Sathe, S., Keedy, R., Stein, K., 2006. Space–time finite element techniques for computation of fluid–structure interactions. Computer Methods in Applied Mechanics and Engineering, 195(17–18), 2002–2027.
[22] Steinbach, O., 2015. Space-Time Finite Element Methods for Parabolic Problems. Computational Methods in Applied Mathematics. 15(4), 551–566.
[23] Tezduyar, T.E., Asada, S., Kuraishi, T., 2016. Space–Time method for flow computations with slip interfaces and topology changes (ST-SI-TC). Computers & Fluids. 141, 124–134.
[24] Feng, L.B., Zhuang, P., Turner, I., Gu, Y.T., 2016. Finite element method for space-time fractional diffusion equation. Numerical Algorithms. 72(3), 749–767.
[25] Bause, M., Radu, F.A., Köcher, U., 2017. Space–time finite element approximation of the Biot poroelasticity system with iterative coupling. Computer Methods in Applied Mechanics and Engineering. 320, 745–768.
[26] Singh, G., Wheeler, M.F., 2018. A space–time domain decomposition approach using enhanced velocity mixed finite element method. Journal of Computational Physics. 374, 893–911.
[27] Zwart, P.J., Raithby, G.D., Raw, M.J., 1999. The Integrated Space-Time Finite Volume Method and Its Application to Moving Boundary Problems. Journal of Computational Physics. 154(2), 497–519.
[28] Klaij, C.M., van der Vegt, J.J.W., van der Ven, H., 2006. Space–time discontinuous Galerkin method for the compressible Navier–Stokes equations. Journal of Computational Physics. 217(2), 589–611.
[29] Netuzhylov, H., Zilian, A., 2009. Space-time meshfree collocation method: Methodology and application to initial-boundary value problems. International Journal for Numerical Methods in Engineering. 80(3), 355–380.
[30] Dohr, S., Zapletal, G., Merta, M., Kravčenko, M., 2019. A parallel space–time boundary element method for the heat equation. Computers & Mathematics with Applications. 78(9), 2852-2866.
[31] Zhang, C., Yao, H., Li, H., 2019. New space–time spectral and structured spectral element methods for high order problems. Journal of Computational and Applied Mathematics. 351, 153–166.

[32] Reinstädler, S., Kowalsky, U., Dinkler, D., 2019. Analysis of landslides employing a space–time single-phase level-set method. Computer Methods in Applied Mechanics and Engineering. 347, 639–662.
[33] Lynn, P.P., Arya, S.K., 1974. Finite elements formulated by the weighted discrete least squares method. International Journal for Numerical Methods in Engineering. 8(1), 71–90.
[34] Fix, G.J., Gunzburger, M.D., Nicolaides, R.A., 1979. On finite element methods of the least squares type. Computers & Mathematics with Applications. 5(2), 87–98.
[35] Jiang, B., 1998. The Least-Squares Finite Element Method. Berlin, Heidelberg: Springer Berlin Heidelberg (Scientific Computation).
[36] Zhang, X., Liu, X., Song, K., Lu, M., 2001. Least-squares collocation meshless method. International Journal for Numerical Methods in Engineering. 51(9), 1089–1100.
[37] Afshar, M.H., Lashckarbolok, M., 2008. Collocated Discrete Least-Squares (CDLS) meshless method: Error estimate and adaptive refinement. International Journal for Numerical Methods in Fluids. 56(10), 1909–1928.
[38] Wang, Q.X., Li, H., Lam, K.Y., 2005. Development of a new meshless - Point Weighted Least-Squares (PWLS) method for computational mechanics. Computational Mechanics. 35(3), 170–181.
[39] Kee, B.B.T., Liu, G.R., Lu, C., 2007. A Regularized Least-Squares Radial Point Collocation Method (RLS-RPCM) for adaptive analysis. Computational Mechanics. 40(5), 837–853.
[40] Hughes, T.J.R., Stewart, J.R., 1996. A space-time formulation for multiscale phenomena. Journal of Computational and Applied Mathematics. 74(1–2), 217–229.
[41] Liu, G.R., 2010. Meshfree methods : moving beyond the finite element method. CRC Press.
[42] Pissanetzky, S., 1984. Sparse matrix technology. Academic Press.
[43] Zienkiewicz, O.C., Taylor, R.L., 2013. Finite element method : its basis and fundamentals. Butterworth-Heinemann.
[44] Jiang, L., Liu, F. and Chen, D., 2015. A fast particle level set method with optimized particle correction procedure for interface capturing. Journal of Computational Physics. 299, 804–819.
[45] Touré, M.K., Soulaïmani, A., 2016. Stabilized finite element methods for solving the level set equation without reinitialization. Computers & Mathematics with Applications. 71(8), 1602–1623.
[46] Zhao, L., Khuc, H., Mao, J., Liu, X., Avital, E., 2018. One-layer particle level set method. Computers & Fluids. 170, 141–156.