INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2002; **39**:277–299 (DOI: 10.1002/fld.324)

Comparison of c-space and p-space particle tracing schemes on high-performance computers: accuracy and performance

F. Schäfer and M. Breuer*,†

Lehrstuhl für Strömungsmechanik (LSTM), Universität Erlangen-Nürnberg, Cauerstr. 4, D-91058 Erlangen, Germany

SUMMARY

The present paper presents a comparison of four different particle tracing schemes which were integrated into a parallel multiblock flow simulation program within the frame of a co-visualization approach. One p-space and three different c-space particle tracing schemes are described in detail. With respect to application on high-performance computers, parallelization and vectorization of the particle tracing schemes are discussed. The accuracy and the performance of the particle tracing schemes are analyzed extensively on the basis of several test cases. The accuracy with respect to an analytically prescribed and a numerically calculated velocity field is investigated, the latter in order to take the contribution of the flow solver's error to the overall error of the particle traces into account. Performance measurements on both scalar and vector computers are discussed. With respect to practical CFD applications and the required performance especially on vector computers, a newly developed, improved c-space scheme is shown to be comparable to or better than the investigated p-space scheme. According to accuracy the new c-space scheme is considerably more advantageous than traditional c-space methods. Finally, an application to a direct numerical simulation of a turbulent channel flow is presented. Copyright © 2002 John Wiley & Sons, Ltd.

KEY WORDS: visualization; multi-phase flows; particle tracing; c-space; p-space; high-performance computers

1. INTRODUCTION

Particle tracing techniques are widely used for the visualization of results in computational fluid dynamics (CFD). Optimization of particle tracing schemes for use on high-performance computers is an important issue in co-visualization, where visualization modules are usually part of the flow simulation program [1, 2]. In contrast to the more common post-visualization approach, in co-visualization the major part of the mapping step (calculation of particle traces) is usually done on a high-performance computer, while the particle data are transferred on-line to a graphics workstation for rendering. In addition to co-visualization, the application

* Correspondence to: M. Breuer, LSTM Erlangen, Universität Erlangen-Nürnberg, Cauerstrasse 4, D-91058 Erlangen, Germany.
† E-mail: breuer@lstm.uni-erlangen.de

Received September 2000
Revised October 2001

of particle tracing methods on supercomputing platforms is also important in the field of multi-phase flow simulations based on the Euler–Lagrangian approach. To yield maximum performance of particle tracing schemes in these application cases, the algorithms have to be adapted to high-performance computing architectures.

In the present investigation, four different particle tracing schemes were integrated into a parallel multiblock flow simulation program within the frame of a co-visualization approach. Since the flow solver is mainly applied on vector and parallel-vector computers, parallelization and vectorization of the particle tracing schemes was an important issue. Although pure vector computers may be a dead-end, newer developments of top-level supercomputers such as Hitachi SR8000-F1 show that vectorization is still an important issue. Cache-based systems typically suffer from a dramatical performance breakdown when the data size exceeds the cache size. Applying so-called pseudo-vector facilities seems to be a way out of this dilemma. Therefore, appropriate vectorization of algorithms will remain of major concern not only for parallel-vector computers but also for SMP-clusters and is addressed here.

The flow simulation program applied in the present work makes use of block-structured curvilinear grids, for which two major particle tracing approaches have to be distinguished: the equation of motion of the particles may be integrated either in computational space (c-space) or in physical space (p-space). C-space and p-space particle tracing schemes have been described by several researchers [3–5] and each approach has its specific advantages and disadvantages. In c-space schemes the particle traces are integrated in a coordinate system, in which the curvilinear physical space grid is orthonormal. Point location within the c-space grid is trivial, since there is an explicit relationship between the c-space coordinates of a point (e.g. a particle location) and the grid cell containing it. In p-space generally there is no such explicit relation, so that some iterative method is required to localize a point within the grid. Since vectorization of the point location is more sophisticated than in the case of a c-space scheme, optimization of p-space schemes for use on vector computers is much more difficult. On the other hand, in c-space schemes there are additional sources of error compared with p-space schemes. By comparing several p-space and c-space schemes, Sadarjoen *et al.* [4] have shown that with respect to a given velocity field c-space schemes are usually less accurate than p-space schemes. Moreover, at least on a workstation, most c-space schemes have been found to be significantly slower than schemes working in p-space.

Within the scope of the present investigation, an improved c-space scheme has been developed which is considerably more accurate and more efficient with respect to the operation count than traditional c-space schemes. A comparison of three different c-space schemes and one p-space scheme with the main focus on high-performance computing platforms is presented here.

The paper is organized as follows. Some background information about this work is given in Section 2. Mathematical details of the particle tracing schemes and the applied high-performance computing techniques such as vectorization and parallelization are described in Section 3. The investigation of accuracy with respect to an analytically prescribed velocity field is presented in Section 4.1. In Section 4.2 the contribution of the numerical error of the flow solver to the overall error of the particle tracing schemes is examined. This is especially important for practical CFD applications. The performance (computational speed) of the schemes was measured both on scalar and vector computers, as will be discussed in Section 5. An application to an extensive direct numerical simulation of a turbulent channel flow is presented in Section 6.

2. BACKGROUND

The work presented here is based on a co-visualization approach using the extracts concept of Globus [6], which has been applied previously by Haimes [7]. Extracts are sets of physical information such as the positions of particles and the values of flow quantities at these positions, which are calculated by visualization modules within the flow simulation program. Using this approach, the data output of the flow solver can be reduced drastically since the extracts need less memory than the storage of the complete flow solution. This is especially important for the visualization of extensive direct numerical simulations (DNS) of turbulent flows typically producing several terabytes of solution data. With this concept DNS visualization can be performed at high temporal resolution, which is hardly possible with a post-processing approach since this would require all time steps of the flow solution to be stored. Moreover, the data reduction also facilitates interactive on-line visualization of concurrent flow simulations, since transmission of extracts over a network connection from a high-performance computer to a graphics workstation is considerably faster than transmission of the complete flow solution. However, a serious drawback of this approach is that the flow simulation has to be repeated if previous time steps are to be re-visualized with different parameters.

In this context, a particle tracing module has been integrated into the general-purpose CFD package FASTEST-3D developed by LSTM Erlangen [8, 9]. With this program laminar as well as turbulent steady and unsteady flows including heat and mass transfer can be simulated numerically. The three-dimensional incompressible Navier–Stokes equations expressing the conservation of mass, momentum and energy are solved based on a fully conservative finite-volume discretization on non-orthogonal curvilinear grids with a colocated arrangement of the variables. In order to resolve complex geometries, block-structured grids are used, i.e. the blocks are globally unstructured, but each block consists of a curvilinear structured grid. Second-order central differences are applied for all terms together with flux blending (first-order upwind/second-order central) and a deferred correction approach for the convective fluxes. For the temporal integration of the flow field a second-order fully implicit scheme and a second-order Crank–Nicolson scheme are available. The code is endowed with fast and efficient numerical algorithms such as FAS/FMG multigrid and highly optimized for high-performance supercomputers such as vector computers and parallel-vector systems (e.g. NEC SX-4/5 or Fujitsu VPP 700). Furthermore, it is adapted to SMP-clusters such as the Hitachi SR8000-F1 using pseudo-vector facilities. For parallelization a grid partitioning technique combined with explicit message passing based on MPI is employed. FASTEST-3D has been applied successfully to a long list of engineering and scientific flow problems ranging from external flows around high-speed trains to internal flows in stirred vessels and nearly everything in between.

3. PARTICLE TRACING METHODS

This section gives a description of four particle tracing schemes for 3D time-dependent flows which were integrated into the flow solver FASTEST-3D. An interesting aspect is that FASTEST-3D makes use of block-structured grids and solves the governing equations of fluid flow in c-space based on precomputed transformation matrices between p-space and c-space. The question arises whether these resources can be utilized in a c-space particle

tracing scheme. The c-space scheme CG using the flow solver's transformation matrices is investigated here. However, a detailed analysis will show that the flow solver and the particle tracing scheme need different transformation matrices, resulting in significant errors of the CG scheme. The c-space scheme CZ described as 'C-FDBD'-scheme in Reference [4], where it was found to be the most accurate c-space scheme investigated, is used for comparison purposes. Major improvements to the CZ scheme with respect to accuracy and performance lead to the new c-space scheme CZ+ which will be described in detail. The state-of-the-art p-space scheme PT will be used as reference for the accuracy and performance measurements in Section 4 and Section 5, respectively.

Since the particle tracing schemes are part of the flow solver, there are some specific benefits as well as limitations which are worth noticing. On the one hand, in multiblock and parallel applications the data structures of the flow solver can be used for the exchange of particles between different blocks or processors. On the other hand, in parallel applications no load balancing of particle tracing is possible due to the distributed memory parallelization of FASTEST-3D based on domain decomposition. Furthermore, due to memory limitations in many application cases the flow solver holds only two consecutive time steps of the flow solution in memory, so that the temporal interpolation of the velocity field needed for particle tracing is limited to second-order linear interpolation. For the same reason, the time step for the integration of the particle paths cannot exceed the simulation time step of the flow solver.

The starting point of this section is the governing equation of particle advection in a flow and its numerical integration, followed by a description of the p-space scheme PT and the three c-space schemes CG, CZ, and CZ+. First the schemes are formulated for the one-block case of a structured curvilinear grid. Multi-block grids are discussed together with the parallelization and vectorization techniques at the end of this section.

3.1. Governing equation and numerical integration

The objective of particle tracing as a flow visualization method is to create an intuitive representation of the velocity field, not to trace real physical objects mutually interacting with the flow. A particle in this sense is massless and infinitely small, always moving with the same velocity as the fluid at the actual particle position, and never acting back on the fluid. A particle trace resulting from this advection process is given by the solution of the following initial value problem:

$$\dot{\mathbf{x}}(t) = \mathbf{v}(\mathbf{x}, t), \quad \mathbf{x}(\mathbf{t}_0) = \mathbf{x}_0 \tag{1}$$

where $\mathbf{x}(t)$ is the particle position at time t, $\mathbf{v}$ denotes the velocity field and $\mathbf{x}_0$ is the initial position of the particle at $t = t_0$.

The main difficulty with the integration of Equation (1) is that the velocity is given as a discrete CFD solution at the vertices of a structured grid at discrete time steps. Some interpolation scheme is needed to construct the velocity field at arbitrary positions $(\mathbf{x}, t)$ within the computational domain, which in turn requires the knowledge of the grid cell containing the point $\mathbf{x}$ at time t. Hence a point location scheme is necessary which relates the coordinates of a given point to the grid cell containing it. Since point location is completely different in c-space and p-space schemes, the details are described later together with the specific schemes.

For the numerical integration of Equation (1), both non-adaptive and adaptive, explicit and linear-implicit Runge–Kutta (RK) schemes [10, 11] have been implemented, e.g. RK2,

RK2(1), RK3, RK3(2) and LIRK4(3), where RKp refers to a non-adaptive explicit RK scheme of order p, RK$p_1(p_2)$ is an adaptive scheme of order p_1, which uses an embedded RK scheme of order p_2 for the error estimation, and LIRK refers to a linear-implicit RK scheme. As an example, two different RK2 schemes will be specified here. In the so-called modified Euler scheme, the following explicit relations are applied to obtain a new particle position $\mathbf{x}_{l+1}$ at time $t+\Delta t$ from a position $\mathbf{x}_l$ at time t:

$$\begin{aligned} a &= \tfrac{1}{2}\,\Delta t \\ \mathbf{b} &= \mathbf{x}_l + a\,\mathbf{v}(\mathbf{x}_l, t) \\ \mathbf{x}_{l+1} &= \mathbf{x}_l + \Delta t\,\mathbf{v}(\mathbf{b}, t+a) \end{aligned} \tag{2}$$

where Δt is the integration time step. Another RK2 scheme, the so-called Heun scheme, can be obtained by substituting $a=\Delta t$ in the above relations. For the integration schemes it is important that the flow solver holds only two consecutive time steps t_{n-1} and t_n $(n=1,2,\ldots)$ of the flow solution in memory, so that the integration time step cannot exceed the simulation time step $t_n - t_{n-1}$ of the flow solver.

So far the governing equation and the numerical integration schemes have been considered in a physical space formulation, which can be applied in a p-space particle tracing scheme. A description of the p-space scheme PT applied in the present investigation is given in the next section. For a c-space scheme, the initial value problem Equation (1) has to be transformed to the c-space coordinate system, as will be described in Section 3.3.

3.2. P-space scheme PT

Since in a general curvilinear grid there is no explicit relation between a given point and the grid cell containing it, in p-space some iterative method has to be used for point location. In this context global and local search algorithms have to be distinguished. If a point in the vicinity of the target point has already been located before, the known point can be used as the starting point for a local search based on popular methods such as stencil walk [4], the Newton–Raphson iterative method [4], or tetrahedral walk [5]. A global search is usually necessary to locate the initial position $\mathbf{x}_0$ of a particle, whereas a local search can be used to locate the subsequent particle positions obtained by means of the respective numerical integration scheme.

In the present investigation a p-space scheme using a tetrahedral walk for local search is applied. As Kenwright and Lane [5] have shown, this search method is considerably more efficient than the Newton–Raphson iterative method. It is based on a tetrahedral decomposition of the hexahedral grid cells, which is done on-the-fly as the search advances through the grid. To ensure a consistent decomposition of adjacent grid cells so that there are no gaps between neighboring tetrahedra and no overlapping regions, the algorithm applied here decomposes a hexahedron into six tetrahedra. The benefit of using tetrahedral cells for point location is that it is easy to test whether a given point lies within a tetrahedron. If the vertices of a regular tetrahedron are designated by $\mathbf{p}_1$, $\mathbf{p}_2$, $\mathbf{p}_3$, and $\mathbf{p}_4$, the difference vectors $\mathbf{p}_2-\mathbf{p}_1$, $\mathbf{p}_3-\mathbf{p}_1$, and $\mathbf{p}_4-\mathbf{p}_1$ are linearly independent, forming a basis of the three-dimensional space. Then any point $\mathbf{x}$ can be written as

$$\mathbf{x}(\eta_1,\eta_2,\eta_3) = \mathbf{p}_1 + (\mathbf{p}_2-\mathbf{p}_1)\eta_1 + (\mathbf{p}_3-\mathbf{p}_1)\eta_2 + (\mathbf{p}_4-\mathbf{p}_1)\eta_3 \tag{3}$$

where η_1, η_2, and η_3 are the coordinates of $\mathbf{x}$ with respect to the basis vectors and the origin $\mathbf{p}_1$. They can be calculated analytically by inverting Equation (3). The point $\mathbf{x}$ is inside the tetrahedron if it is surrounded by its four planar faces given by the equations $\eta_1 = 0$, $\eta_2 = 0$, $\eta_3 = 0$, and $\eta_1 + \eta_2 + \eta_3 = 1$, respectively. This is the case if the coordinates of $\mathbf{x}$ meet the following relations:

$$\eta_1 \geqslant 0, \ \eta_2 \geqslant 0, \ \eta_3 \geqslant 0, \quad \text{and} \quad 1 - \eta_1 - \eta_2 - \eta_3 \geqslant 0 \tag{4}$$

If one or more of these conditions are violated, the search proceeds to an adjacent tetrahedron across the face with the strongest violation. This is repeated until the tetrahedron containing the point has been found.

For global point location an algorithm is applied which determines a grid node in the vicinity of the target point $\mathbf{x}$, which is used as a starting point for a local refinement search based on a tetrahedral walk. Given a structured grid with N_i, N_j, and N_k grid cells in the three index directions, the global search starts at the center of the grid at a grid node roughly addressed by the indices $N_i/2$, $N_j/2$, $N_k/2$, searching with adaptive strides in all index directions for a grid node next to the target point. In multiblock grids consisting of several interconnected structured grids (blocks), the global search algorithm has to search every block until the target point is found. The process is accelerated by checking first whether the target point lies within the bounding box of the current block.

Once a position $\mathbf{x}$ is located within the grid, linear interpolation in space and in time is used to construct the velocity $\mathbf{v}(\mathbf{x}, t)$ needed by the numerical integration scheme. Given the flow solution at discrete time steps t_n, $n = 1, 2, 3, \ldots$ at the vertices of the tetrahedron containing $\mathbf{x}$, a general linear interpolation operator $L_{q,n}$ for some flow field quantity q can be defined as

$$\begin{aligned} q(\mathbf{x}, t_n) &= L_{q,n}(\mathbf{x}) \\ &= q_{1,n} + (q_{2,n} - q_{1,n})\eta_1 + (q_{3,n} - q_{1,n})\eta_2 + (q_{4,n} - q_{1,n})\eta_3 \end{aligned} \tag{5}$$

where $q_{m,n}$ refers to the value of q at the vertex $\mathbf{p}_m$ at time t_n. The interpolation factors η_1, η_2, and η_3 are the coordinates of $\mathbf{x}$ according to Equation (3), and as a by-product of point location they need not to be calculated separately. Then for $t \in [t_{n-1}, t_n]$ the velocity interpolation is given by

$$\mathbf{v}(\mathbf{x}, t) = \tau \cdot L_{\mathbf{v},n}(\mathbf{x}) + (1 - \tau) \cdot L_{\mathbf{v},n-1}(\mathbf{x}) \tag{6}$$

where $\tau = [(t - t_{n-1})/(t_n - t_{n-1})]$.

3.3. C-space schemes CG, CZ, and CZ+

The objective of tracing particles in c-space is to make the location of a point within the grid easier than in p-space. Basically the c-space is a coordinate system in which the curvilinear physical space grid is orthonormal, so that there is an explicit relation between the coordinates of some point and the grid cell containing it. As indicated in Figure 1, the transformation of a given curvilinear p-space grid with grid nodes $\mathbf{x}_{i,j,k}$ to an orthonormal c-space grid can be done by mapping the physical coordinates of the nodes to their index vectors $(i, j, k)^t$. In the other direction, the entire transformation T of a given c-space point $\boldsymbol{\xi} = (\xi_1, \xi_2, \xi_3)^t$ to p-space

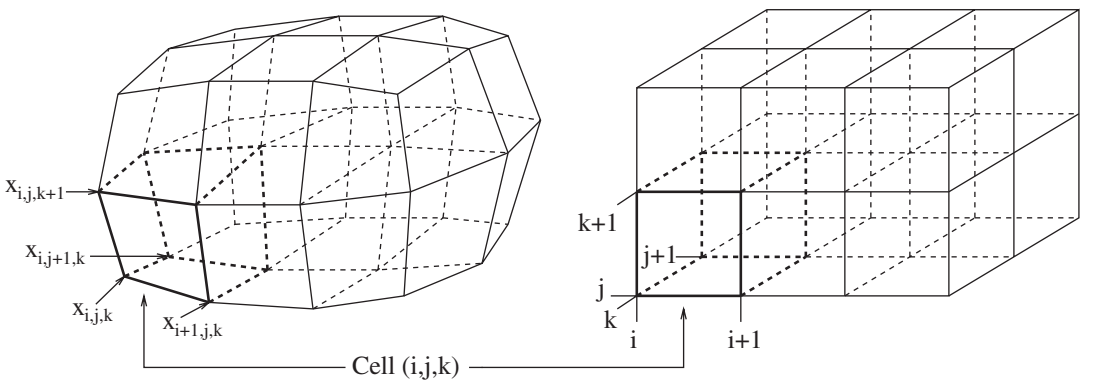


Figure 1. Transformation of a structured curvilinear p-space grid (left) to an orthonormal c-space grid (right).

can be defined by a trilinear interpolation. Decomposing $\boldsymbol{\xi}$ into an integer part $(i,j,k)^t$ and a fractional offset $(\Delta\xi_1,\Delta\xi_2,\Delta\xi_3)^t$,

$$\boldsymbol{\xi}=(i,j,k)^t+(\Delta\xi_1,\Delta\xi_2,\Delta\xi_3)^t,\quad \Delta\xi_m\in[0,1[,\ m\in\{1,2,3\} \tag{7}$$

the transformation can be written as

$$\begin{aligned}\mathbf{x}&=T(\boldsymbol{\xi})\\&=\Delta\xi_3\Delta\xi_2\Delta\xi_1\mathbf{x}_{i+1,j+1,k+1}+\Delta\xi_3\Delta\xi_2(1-\Delta\xi_1)\mathbf{x}_{i,j+1,k+1}\\&\quad+\Delta\xi_3(1-\Delta\xi_2)\Delta\xi_1\mathbf{x}_{i+1,j,k+1}+\Delta\xi_3(1-\Delta\xi_2)(1-\Delta\xi_1)\mathbf{x}_{i,j,k+1}\\&\quad+(1-\Delta\xi_3)\Delta\xi_2\Delta\xi_1\mathbf{x}_{i+1,j+1,k}+(1-\Delta\xi_3)\Delta\xi_2(1-\Delta\xi_1)\mathbf{x}_{i,j+1,k}\\&\quad+(1-\Delta\xi_3)(1-\Delta\xi_2)\Delta\xi_1\mathbf{x}_{i+1,j,k}+(1-\Delta\xi_3)(1-\Delta\xi_2)(1-\Delta\xi_1)\mathbf{x}_{i,j,k}\end{aligned} \tag{8}$$

The point location scheme is represented by Equation (7), relating the c-space point $\boldsymbol{\xi}$ to the index triple $(i,j,k)^t$ which addresses the grid cell containing $\boldsymbol{\xi}$. It can be implemented using integer operators provided by programming languages such as Fortran and C.

Since Equation (8) cannot be inverted analytically, iterative methods have to be applied to determine the c-space position $\boldsymbol{\xi}$ of an arbitrary p-space point $\mathbf{x}$. It is important to note that the transformation is not properly (i.e. uniquely) defined in irregular grid cells which are not hexahedra in p-space, e.g. in cells with one or more edges of zero length. Such irregular cells have to be excluded from the particle tracing domain. However, the same applies to the computational domain of the flow solver used here. Therefore, this condition does not lead to an additional restriction.

By differentiating Equation (8) with respect to time it can be found that a p-space velocity $\mathbf{v}$ given at a position $\mathbf{x}=T(\boldsymbol{\xi})$ is transformed to c-space according to

$$\mathbf{v}(\boldsymbol{\xi},t)=J^{-1}(\boldsymbol{\xi})\mathbf{v}(\mathbf{x},t) \tag{9}$$

where $\mathbf{v}$ refers to the c-space velocity and J is the Jacobian matrix of T. The mth column of J is given by

$$J_m = \frac{\partial \mathbf{x}}{\partial \xi_m} = \frac{\partial T(\boldsymbol{\xi})}{\partial \xi_m} \tag{10}$$

Here it is assumed that the positions $\mathbf{x}_{i,j,k}$ of the grid nodes do not change in time, so that J is independent of time. It should be noted that for curvilinear p-space grids, J is usually discontinuous at grid cell faces, so that the c-space velocity $\mathbf{v}$ is discontinuous at cell faces as well.

Transformation of the initial value problem (Equation (1)) to c-space can be done by multiplying the differential equation from the left with $J^{-1}(\boldsymbol{\xi})$, where $\mathbf{x} = T(\boldsymbol{\xi})$, and choosing the corresponding initial values. This yields

$$\dot{\boldsymbol{\xi}}(t) = \quad \mathbf{v}(\boldsymbol{\xi}, t), \quad \boldsymbol{\xi}(t_0) = \boldsymbol{\xi}_0, \quad \mathbf{x}_0 = T(\boldsymbol{\xi}_0) \tag{11}$$

$\boldsymbol{\xi}_0$ is determined from the given p-space position $\mathbf{x}_0$ by means of an iterative search algorithm which consists of a global search to find a grid node next to $\mathbf{x}_0$ and a subsequent local refinement based on Newton–Raphson iterations [4].

For the construction of the velocity field $\mathbf{v}(\boldsymbol{\xi}, t)$ at arbitrary positions, a trilinear spatial and linear temporal interpolation is applied. In the following it is assumed that the flow solution is provided either at the grid nodes $\mathbf{x}_{i,j,k}$ or at the centres of the grid cells. Then the c-space positions $\boldsymbol{\zeta}_{i,j,k}$ of the points where the flow solution is supplied can be written as

$$\boldsymbol{\zeta}_{i,j,k} = (i, j, k)^t + \mathbf{s} \tag{12}$$

with $\mathbf{s} = 0$ (flow field quantities given at the grid nodes) or $\mathbf{s} = 1/2(1, 1, 1)^t$ (quantities given at the centres of the grid cells), respectively. The points $\boldsymbol{\zeta}_{i,j,k}$ can be viewed as the nodes of a second grid which will be referred to as the $\boldsymbol{\zeta}$-grid in the following. Given the flow solution at discrete time steps t_n, $n = 1, 2, 3, \ldots$, a general trilinear interpolation operator $T_{q,n}$ for some flow field quantity q can be defined, so that

$$q(\boldsymbol{\xi}, t_n) = T_{q,n}(\boldsymbol{\xi}'), \quad \boldsymbol{\xi}' = \boldsymbol{\xi} - \mathbf{s} \tag{13}$$

$T_{q,n}$ is defined similarly to the coordinate transformation T in Equation (8), except that the terms of form $\mathbf{x}_{i,j,k}$ are replaced by $q_{i,j,k,n}$, where the latter is the value of q at the point $\boldsymbol{\zeta}_{i,j,k}$ at time t_n. Then for $t \in [t_{n-1}, t_n]$ the velocity interpolation is given by

$$\mathbf{v}(\boldsymbol{\xi}, t) = \tau \cdot T_{\mathbf{v},n}(\boldsymbol{\xi}') + (1 - \tau) \cdot T_{\mathbf{v},n-1}(\boldsymbol{\xi}') \tag{14}$$

where $\tau = [(t - t_{n-1})/(t_n - t_{n-1})]$. This requires the c-space velocities $\mathbf{v}_{i,j,k,n}$ to be calculated, which can be obtained from the p-space velocities $\mathbf{v}_{i,j,k,n}$ provided by the flow solver by applying Equation (9). Since this has to be done for eight nodes at two time steps, 16 velocity transformations have to be performed.

Trilinear interpolation of the c-space velocity $\mathbf{v}$ is very critical. If in a first-order approximation the p-space velocity $\mathbf{v}$ is assumed to be linear along the cell edges of the $\boldsymbol{\zeta}$-grid, the transformed c-space velocity is only linear if the Jacobian of the transformation is constant along the edges (compare Equation (9)). For $\mathbf{s} = 0$ this is the case if the grid cells are

parallelepipeds, but generally it is not. Therefore, trilinear interpolation of the c-space velocity as it is used in Equation (14) can lead to considerable interpolation errors. As an alternative, the p-space velocity $\mathbf{v}(\mathbf{x},t)$ at the position $\mathbf{x}=T(\xi)$ can be interpolated by using

$$\mathbf{v}(\mathbf{x},t)=\tau\cdot T_{\mathbf{v},n}(\xi')+(1-\tau)\cdot T_{\mathbf{v},n-1}(\xi') \tag{15}$$

The velocity $\mathbf{v}(\mathbf{x},t)$ can be transformed to c-space according to Equation (9) by using the Jacobian $J(\xi)$ at the query point. Because only one velocity transformation has to be performed instead of 16, this approach leads to a considerably smaller operation count. Moreover, applying Equation (15) instead of Equation (14) results in significantly smaller errors of the c-space particle tracing scheme, as will be shown in Section 4.

For the numerical integration of the initial value problem given by Equation (11), the RK methods described in Section 3.1 are applied accordingly. For example, in Equation (2) the p-space coordinates $\mathbf{x}$ and velocities $\mathbf{v}$ have to be replaced by the c-space coordinates ξ and velocities $\mathbf{v}$, respectively.

In this paper two different coordinate transformations from p-space to c-space are considered, which are referred to as TG and TZ. Details of these transformations are given in the next section.

3.3.1. Coordinate transformations TG and TZ. In the present investigation the particle tracing schemes are part of a finite-volume flow solver which is based on a colocated arrangement of the variables. In this context two different coordinate transformations from p-space to c-space have to be considered. Since the flow solver is operating in c-space, there is a built-in transformation which is depicted in Figure 2. The finite-volume method operates on the cells of the **g**-grid (solid lines, grid nodes $\mathbf{g}_{i,j,k}$ in 3D) and provides the flow solution at the centers $\mathbf{z}_{i,j,k}$ of the cells. To obtain control volumes which are unit cubes in c-space, the built-in transformation maps the **g**-grid to an orthonormal grid. Formally this can be achieved

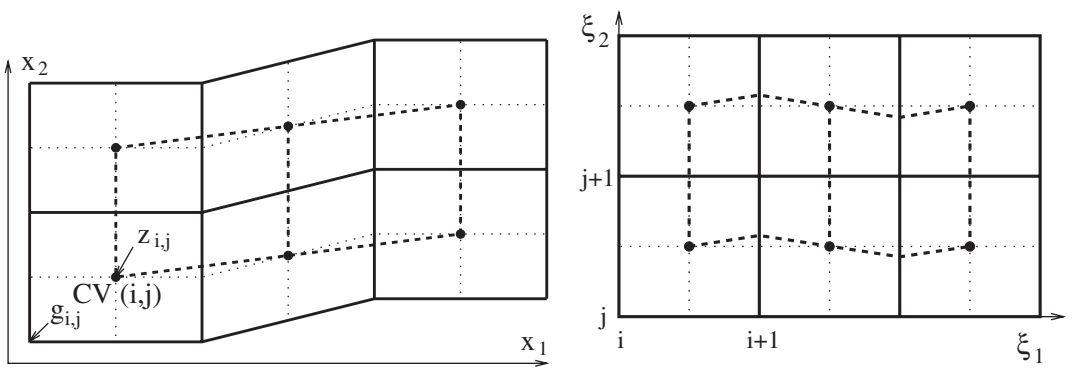


Figure 2. Transformation of a curvilinear p-space grid (left, solid lines) to an orthonormal c-space grid (right), as performed in the flow solver applied in the present investigation (2D schematic drawing). The cells of the **g**-grid (solid lines, grid nodes $\mathbf{g}_{i,j}$) are the control volumes (CV) on which the finite-volume flow solver operates. The flow solution is provided at the centers $\mathbf{z}_{i,j}$ of the **g**-grid cells, which constitute the **z**-grid given by the thick dashed lines. While the **g**-grid is orthonormal in this c-space, the **z**-grid is not.

by setting $\mathbf{x}_{i,j,k} = \mathbf{g}_{i,j,k}$ in Equation (8). In the following this transformation is designated by TG.

Since the velocity is provided at the centres $\mathbf{z}_{i,j,k}$ of the control volumes, the **z**-grid (thick dashed lines, grid nodes $\mathbf{z}_{i,j,k}$) is more relevant for particle tracing than the **g**-grid. Unfortunately, the transformation TG does not map the **z**-grid to an orthonormal grid in c-space, leading to significant errors when applying trilinear velocity interpolation. To avoid these problems, a second transformation TZ is considered which maps the **z**-grid to an orthonormal grid. This transformation can be obtained by setting $\mathbf{x}_{i,j,k} = \mathbf{z}_{i,j,k}$ in Equation (8). It is important to note that $\mathbf{s} = 0$ in Equation (12) for TZ, whereas $\mathbf{s} = 1/2(1,1,1)^t$ for TG. For both transformations the c-space point $\zeta_{i,j,k}$ where the flow solution is supplied corresponds to the **z**-grid node $\mathbf{z}_{i,j,k}$ in p-space.

Three different c-space particle tracing schemes (CG, CZ, CZ+) based on the transformations TG and TZ are described in the following sections. While the scheme CG applies the built-in transformation TG of the flow solver, the schemes CZ and CZ+ are based on TZ.

3.3.2. C-space scheme CG. Since the flow solver is operating in c-space, precomputed transformation matrices J^{-1} are available for TG. The matrices are supplied at the **z**-grid nodes, based on an analytical evaluation of Equation (10) at the corresponding c-space positions $\zeta_{i,j,k}$, e.g.

$$J_1(\zeta_{i,j,k}) = \mathbf{f}_{i+1} - \mathbf{f}_i$$
$$\mathbf{f}_i = \tfrac{1}{4}(\mathbf{g}_{i,j+1,k+1} + \mathbf{g}_{i,j,k+1} + \mathbf{g}_{i,j+1,k} + \mathbf{g}_{i,j,k})$$

In order to utilize these precomputed matrices for particle tracing, the scheme CG based on the transformation TG was investigated. For construction of the velocity field an interpolation of the c-space velocity according to Equation (14) is applied. Since the precomputed matrices are supplied at the **z**-grid nodes only, interpolation of the p-space velocity according to Equation (15) is not used because this requires the transformation matrices $J(\xi)$ at arbitrary positions ξ within the grid.

3.3.3. C-space scheme CZ. CZ is identical with the 'C-FDBD'-scheme described in Reference [4], where it was found to be the most accurate c-space scheme investigated there. The scheme is based on the transformation TZ, and interpolation of the c-space velocity according to Equation (14) is applied for the construction of the velocity field. This requires the transformation matrices to be calculated at the **z**-grid nodes, which is done on-the-fly for all eight vertices of the grid cell containing the current query position ξ. The matrices are obtained by evaluating Equation (10) at the c-space positions $\zeta_{i,j,k}$ corresponding to the **z**-grid nodes. It is important to note that the evaluation of Equation (10) at $\zeta_{i,j,k}$ is ambiguous since the derivatives may be discontinuous. Right- or left-sided limits of the derivatives are used depending on the location of the query position ξ with respect to the current grid node. Decomposing ξ into $\zeta_{i,j,k}$ (which is identical with $(i,j,k)^t$ for TZ) and a fractional offset according to Equation (7) leads to forward/backward differences such as

$$J_1(\zeta_{i,j,k}) = \mathbf{z}_{i+1,j,k} - \mathbf{z}_{i,j,k}$$
$$J_1(\zeta_{i+1,j,k}) = \mathbf{z}_{i+1,j,k} - \mathbf{z}_{i,j,k}$$

Table I. Summary of important properties of the investigated particle tracing schemes. The transformation related entries are relevant only for the c-space schemes and refer to the computational resources needed for the calculation of the c-space velocity at one query position. The operation count refers to the number of floating point operations required to determine the p-space or c-space velocity at one query position. For the p-space scheme this depends on the number of iterations I ($\geqslant 1$) needed to perform a local query point location.

Coordinate system	p-space	c-space		
Scheme	PT	CG	CZ	CZ+
Point location	iterative	explicit	explicit	explicit
Transformation	none	TG (**g**-grid)	TZ (**z**-grid)	TZ (**z**-grid)
Transformation matrices	none	8, precomputed at **z**-grid nodes	8, computed at **z**-grid nodes	1, computed at query position
Velocity transformations	none	16	16	1
Operation count	$60 + I \cdot 70$	437	729	309
Construction of velocity field	interpolation of p-space velocity (Equation (6))	interpolation of c-space velocity (Equation (14))	interpolation of c-space velocity (Equation (14))	interpolation of p-space velocity (Equation (15))
Vectorization	−	+	+	+
Parallelization	(+)	+	+	+

3.3.4. C-space scheme CZ+. The scheme CZ+ is an enhancement of CZ, applying interpolation of the p-space velocity according to Equation (15) for the construction of the velocity field. This requires the transformation matrices to be calculated at arbitrary query positions within the grid. Evaluating Equation (10) analytically, the first column of the Jacobian at some position $\boldsymbol{\xi}$ is found to be

$$\begin{aligned} J_1(\boldsymbol{\xi}) = {} & \Delta\xi_3\Delta\xi_2(\mathbf{z}_{i+1,j+1,k+1} - \mathbf{z}_{i,j+1,k+1}) \\ & + \Delta\xi_3(1-\Delta\xi_2)(\mathbf{z}_{i+1,j,k+1} - \mathbf{z}_{i,j,k+1}) \\ & + (1-\Delta\xi_3)\Delta\xi_2(\mathbf{z}_{i+1,j+1,k} - \mathbf{z}_{i,j+1,k}) \\ & + (1-\Delta\xi_3)(1-\Delta\xi_2)(\mathbf{z}_{i+1,j,k} - \mathbf{z}_{i,j,k}) \end{aligned}$$

where $\boldsymbol{\xi}$ has to be decomposed according to Equation (7).

The scheme CZ+ leads to smaller errors than CZ, since it avoids the trilinear interpolation of the c-space velocity. Moreover, as shown in Table I, the operation count for CZ+ is considerably smaller than for CZ and even CG, although the latter makes use of precomputed Jacobians.

3.4. High-performance computing techniques

The above particle tracing schemes were integrated into the flow solver FASTEST-3D, which is to a high degree optimized for usage on vector and parallel-vector computers. Hence the particle tracing schemes need to be adapted to these platforms as well. Some details of the

strategies which were applied for the parallelization and vectorization of the particle tracing schemes are given in the following.

3.4.1. Parallelization. Within FASTEST-3D, block-structured curvilinear grids are used in order to resolve complex geometries. Parallelization of FASTEST-3D is based on a grid partitioning approach, i.e. different blocks of the grid can be assigned to different processors. A distributed memory parallelization is applied, so that processors cannot access the data of other processors directly. Explicit message passing based on MPI is used to exchange data between processors whenever necessary.

Within each single block particle tracing takes place in a structured curvilinear grid, and the overlying block structure has only to be taken into account when particles are moving from one block to another. To identify the block in which a particle is currently moving, each particle carries the current block number as an additional attribute besides the particle position. A domain check is performed at the end of each integration time step to check whether a particle has passed the interface between two blocks. In this case particles have to be exchanged between the blocks and, if necessary, between the corresponding processors. Here the parallel structure of FASTEST-3D can be used to its full extent. When a particle leaves a block, no search of the destination block has to be performed since it is known *a priori* from look-up tables. Furthermore, in the case of the c-space schemes no search of the particle's new c-space position is necessary since a well defined coordinate transformation can be applied to obtain the c-space position in the new block from the position in the previous one. For the p-space scheme at least a grid node in the vicinity of the new particle position in the neighboring block is known *a priori*, so that a local tetrahedra-based search is sufficient for point location. A time-consuming global point location is not necessary.

Owing to the distributed memory parallelization of FASTEST-3D, no load balancing of particle tracing is possible. Since there is no direct access especially to the grid and the flow field data of the other processors, each processor can integrate only the traces of the particles which are currently in one of its own blocks. However, since the computational time for particle tracing is typically small compared with the entire CPU time of the flow solver, this approach is acceptable.

3.4.2. Vectorization. To facilitate vectorization of the particle tracing schemes, the attributes of the particles such as position and block number are organized in arrays (one array for each attribute, with the attribute value of the current mth particle as the mth element of the array). Particle path integration and domain check are done in loops over all particles of a processor, and the main task is to implement these loops in such a way that there are no data dependences inhibiting vectorization. This is straightforward for the c-space schemes, which is mainly due to the vectorizable point location given by Equation (7).

The major problem with the vectorization of the p-space scheme is the iterative point location which is based on a tetrahedral walk. In an algorithm optimized for scalar computers, it is reasonable to locate each particle completely before location of the next one is started, whereas for vector computers it is better to do a single iteration of the point location in a loop over all particles, and to repeat this loop until all particles are completely located. Before each iteration a list of the particles which are not yet located is set up or updated, so that the loop can be restricted to the particles which are not yet finished (indirect addressing of the particle arrays). This results in an algorithm which is considerably more sophisticated than in

the scalar case. Besides an array for the particle list, several additional arrays corresponding in size to the number of particles are needed for auxiliary purposes. In this way vectorization of the iterative point location is achieved, although it is less efficient owing to use of indirect addressing and considerably more memory resources are required. Additionally, since the vector and the scalar algorithm are to a high degree adapted to the respective platform, two different algorithms need to be implemented to obtain a p-space scheme which is optimized for both scalar and vector computers (compare the performance results in Section 5).

4. INVESTIGATION OF ACCURACY

There are several sources of error in particle tracing, limiting the accuracy of the resulting particle traces. Common errors of both p-space and c-space particle tracing schemes result from velocity interpolation, numerical integration of the equation of motion, and flow solver accuracy in the case of a numerically calculated velocity field. In this investigation a linear (p-space scheme) or trilinear (c-space schemes) spatial interpolation combined with linear temporal interpolation is applied, leading to an approximation of second order. Higher-order interpolation schemes have been proposed [12, 13], especially for the visualization of turbulent flows calculated with a spectral simulation code. However, in the present investigation particle tracing is limited to second-order linear interpolation in time, since it takes place within a flow solver which holds only two consecutive time steps of the flow solution in memory. With respect to the accuracy impact of the numerical integration scheme, we refer to a discussion elsewhere [14, 15].

Generally in c-space schemes, the following *additional* sources of error compared with p-space schemes can be distinguished.

- Numerical approximation of the transformation matrices.
- Trilinear interpolation of the c-space velocity (see the discussion of Equation (15)).
- Discontinuity of the c-space velocity at cell faces (see the comment on Equation (9)).

Since all three c-space schemes CG, CZ, and CZ+ make use of the exact transformation matrices based on an analytical evaluation of Equation (10), apart from rounding errors there are no errors due to numerical approximation of the matrices. Interpolation of the c-space velocity is only a problem in the schemes CG and CZ, whereas in CZ+ this source of error is eliminated by applying interpolation of the p-space velocity. However, the discontinuity of the c-space velocity at cell faces is a remaining problem in all three c-space schemes. In addition, the c-space scheme CG suffers from the fact that the **g**-grid instead of the **z**-grid (on which the velocity interpolation takes place) is transformed to an orthonormal c-space grid, leading to additional interpolation errors on distorted p-space grids.

To verify these predictions derived from theory, the accuracy of the p-space and c-space particle tracing schemes described in Section 3 was examined extensively on the basis of simple test cases. The test cases were designed to be close to practical CFD applications in order to get an estimate of the overall particle tracing accuracy in real application cases. In particular, no disjoined investigations on the spatial and temporal errors in the limit of very small time steps and grid cell spacings were carried out.

In the first test case an analytically prescribed velocity field was applied in order to monitor the errors of the numerically calculated particle traces with respect to an analytical solution of the equation of motion. These investigations were performed for different modes and degrees of disturbance of an originally cubic grid. Since with respect to practical CFD applications it is important to investigate the contribution of the flow solver's error to the overall error of the particle traces, a second test case was set up using a numerically calculated velocity field. Such investigations do not allow to split up the resulting error in its components owing to the temporal integration of the particle paths, the interpolation error in space and time, and the error of the flow solver. However, for practical applications this decomposition is of minor importance. What counts is the measure of the total error.

4.1. Accuracy for analytically prescribed velocity field

4.1.1. Test case. Within a cubic grid (edge length 0.2 m, 32^3 control volumes), a rotational velocity field $\mathbf{v}(\mathbf{x})=\boldsymbol{\omega}\times\mathbf{x}$ with angular velocity $\boldsymbol{\omega}=(0,0,0.025\ \mathrm{rad\ s^{-1}})^t$ was prescribed, so that an analytical solution of Equation (1) leads to closed circles in the x_1-x_2-plane as particle traces. By determining the difference $\Delta\mathbf{x}=\mathbf{x}_{\mathrm{num}}-\mathbf{x}_{\mathrm{ana}}$ between the numerical and the analytical particle position, the numerical error of the particle tracing schemes was monitored.

The starting point of the calculations was a sampling of up to 2000 particles distributed randomly over the grid. During particle tracing at each instant in time statistical information about the numerical error was recorded. In this way a time series of the average numerical error $\bar{\varepsilon}$ and its estimated standard deviation σ was obtained for every space direction. Assuming that the errors are more or less given by a Gaussian distribution, the errors for the majority of the particles are within the interval $\bar{\varepsilon}\pm\sigma$. To characterize each time series by a single number, the maximum of $|\bar{\varepsilon}\pm\sigma|$ over all time steps was determined, which in the following is referred to as the 'absolute error' of the particle traces.

Several parameter studies were carried out to investigate the different sources of error in detail. In particular, the following parameters were examined.

- Grid disturbance: As shown in Figure 3, starting from the orthonormal grid two disturbance algorithms were applied to yield curvilinear grids with different degrees of disturbance. One method uses trigonometric functions to distort the **g**-grid, resulting in a smooth disturbance. On the other hand, the second method shifts the grid nodes randomly. Both types of disturbances were chosen in order to obtain curvilinear grids with properties typically found in practical CFD applications.

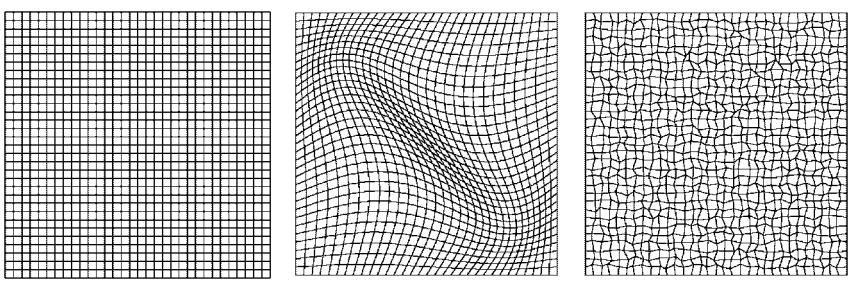


Figure 3. Trigonometric (middle) and random (right) disturbance of an originally cubic grid (left). The largest degree of disturbance used is shown.

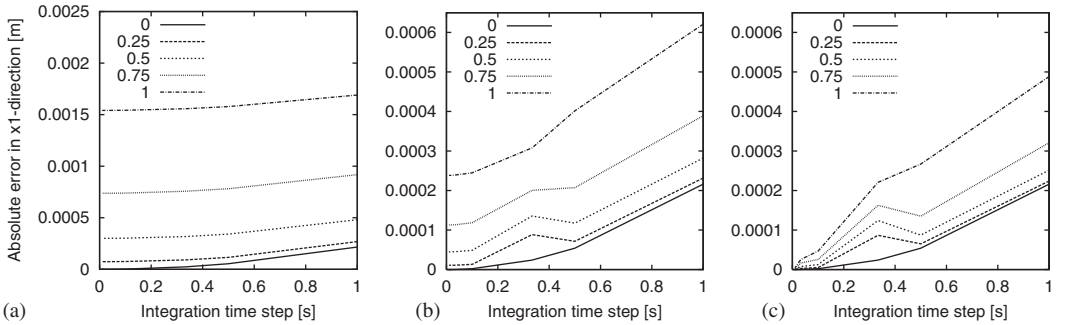


Figure 4. Absolute error of the calculated particle traces for the schemes CG, CZ, and CZ+ (from left to right) using the integrator RK2 and different degrees of *trigonometric grid disturbance*. Disturbance in arbitrary units, where zero refers to the cubic grid. Different error scale in the left diagram. The curves for zero disturbance are identical for all three diagrams.

- Particle tracing schemes: c-space schemes CG, CZ, CZ+; p-space scheme PT.
- Integration schemes: Runge–Kutta integrators RK2 (modified Euler scheme), RK3.
- Integration time step size: To determine an *experimental order of convergence* (EOC), a series with different time step sizes was calculated.

4.1.2. Results. In the following the absolute errors of the particle traces in the x_1-direction are discussed. The results for the x_2-direction are very similar. In the x_3-direction the error is in the range of the machine accuracy, since the velocity is zero in this direction.

Figure 4 shows the results for the three c-space schemes applied together with the RK2 integration scheme and a trigonometric grid disturbance. For the cubic grid the particle tracing errors are dominated by the error of the RK2 scheme (bottom curves in the diagrams), since the EOC value in the investigated time step range meets almost exactly the theoretical order of 2. Accordingly, for the integration scheme RK3 an EOC of almost exactly 3 was obtained on the cubic grid. However, with increasing grid disturbance the errors increase considerably. This behaviour depends strongly on the c-space scheme applied. By far the largest errors are found for CG, whereas the errors are considerably smaller for CZ and CZ+. Moreover, there is a serious convergence problem in the CG scheme, since the errors hardly decrease on going to smaller time steps. Using the RK3 integration scheme, it has even been observed that the absolute error of CG is constant in the investigated time step range. A similar problem occurs in the CZ scheme. Although the overall error is considerably smaller and decreases more distinctively on going to smaller time steps, it does not tend to zero as soon as some grid disturbance is present. This convergence problem is eliminated in the CZ+ scheme, which shows a strong reduction of the error continuing even for the smallest time steps investigated. The EOC values for CZ+ range from 1.35 to 0.98 for low and high degrees of grid disturbance, respectively. Additionally, CZ+ shows significantly smaller overall errors than CZ.

As shown in Figure 5, the results for the random grid disturbance are qualitatively very similar to those obtained for the trigonometric case. Again CG is the scheme with the largest errors, whereas CZ+ is by far the most accurate c-space scheme. However, the overall errors are larger than in the trigonometric case. This is due to the smoothness of the curvilinear

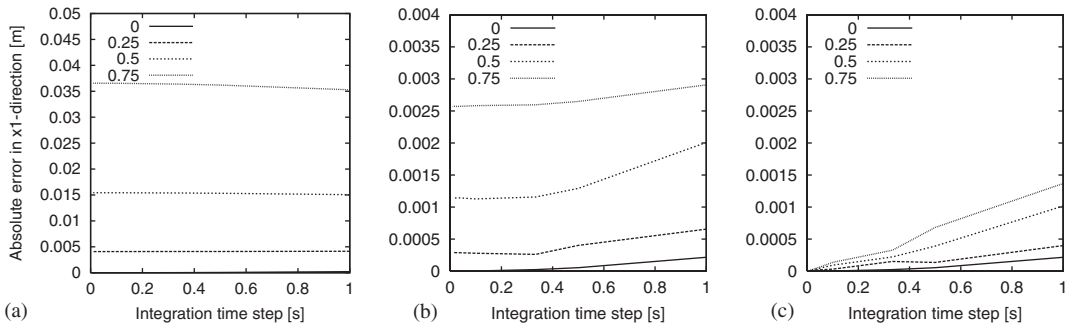


Figure 5. Absolute error of the calculated particle traces for the schemes CG, CZ, and CZ+ (from left to right) using the integrator RK2 and different degrees of *random grid disturbance*. Different error scale in the left diagram.

grids resulting from the trigonometric disturbance, whereas the random disturbance leads to stronger discontinuities of the c-space velocity.

Obviously the trilinear interpolation of the c-space velocity used in the CG and CZ schemes is a major source of error, since the main difference between CZ and CZ+ is that the latter applies interpolation of the p-space velocity. Furthermore, the large errors of CG on distorted grids are in accordance with the presumption that the transformation of the **g**-grid instead of the **z**-grid to an orthonormal grid leads to considerable interpolation errors.

A remaining problem with the CZ+ scheme is that the velocity field in c-space is usually discontinuous at cell faces, which causes the disturbance-induced error shown in the corresponding diagrams. In order to investigate this issue in more detail, the test case was modified so that particle tracing takes place in only one large distorted grid cell. No velocity values from outside the cell were used for particle path integration, thereby avoiding the discontinuity problem. In this case the errors for CZ+ are almost exactly the same as for the p-space scheme PT, which were found to be independent of the grid disturbance (see below). The slight difference is probably due to the different interpolation schemes used in CZ+ and PT (trilinear versus linear). Obviously, the disturbance-induced errors in the original test case are due to the inability of the integration schemes to capture the discontinuity of the velocity field. As a conclusion, a further enhancement of CZ+ could be achieved if it is possible to find an integration scheme which is able to handle the discontinuity better than the Runge–Kutta schemes used here (e.g. some analytical integration method).

The results for the p-space scheme PT are not shown in Figures 4 and 5, since for any kind of grid disturbance the errors for PT are almost exactly the same as for the c-space schemes in the case of a cubic grid (parameter zero in the diagrams). In the investigated time step range the error in PT is clearly dominated by the integration scheme used. Hence, with respect to a given velocity field the p-space scheme is far more accurate than the investigated c-space schemes. However, since the velocity field was not computed by the flow solver, the numerical error of the flow simulation is not included in the error of the particle traces. Hence the accuracy of PT and CZ+ for a numerically calculated velocity field is discussed in the next section.

4.2. *Accuracy for numerically computed velocity field*

4.2.1. Test case. To investigate the contribution of the flow solver's error to the overall error of the particle traces, a lid-driven cavity was considered as a simple test case. The stationary velocity field in a cavity with the same dimensions as in the previous test case and a lid velocity of $0.0025\,\mathrm{m\,s^{-1}}$ was calculated with FASTEST-3D by using an orthogonal computational grid with 64^3 control volumes. The grid is non-equidistant with a stretching factor of 1.002–1.089 in order to yield a higher density of grid cells next to the walls. A trigonometric grid disturbance similar to the previous test case was applied to examine the influence of non-orthogonal grids. The resulting rotational velocity field is similar to the rotational velocity field used in the previous test case.

Since there is no analytical solution to this three-dimensional flow problem, no analytical particle traces can be used to determine the error of the particle tracing schemes. Instead, the particle tracing error was estimated by using a reference flow solution on a very fine orthogonal grid with 128^3 control volumes. Based on this reference solution, reference particle traces were obtained by applying the p-space scheme PT together with the linear-implicit integrator LIRK4(3). By determining the difference $\Delta\mathbf{x} = \mathbf{x}_{\mathrm{num}} - \mathbf{x}_{\mathrm{ref}}$ between the numerical particle position calculated on the coarser grid and the corresponding reference particle position, the particle tracing error was estimated.

Statistical information about the numerical error of up to 2000 particles was recorded over the same period of time as in the previous test case. Again, the resulting time series of the average numerical error and the estimated standard deviation were used to determine the absolute error of the particle traces (compare Section 4.1.1).

These investigations were performed for the c-space scheme CZ+ and the p-space scheme PT. Since the c-space schemes CG and CZ are considerably less accurate and less efficient than CZ+, it is not necessary to consider these two schemes here. Both the integration schemes RK2 (modified Euler scheme) and RK3 were applied to calculate a series of different integration time steps.

4.2.2. Results. While in the previous test case the errors of the p-space scheme were found to be independent of the grid disturbance, Figure 6 clearly shows that in the present test case there is a distinct grid dependence of the particle traces regardless of the particle tracing scheme in use. Neither the results of the CZ+ and PT schemes nor of the integration schemes RK2 and RK3 differ significantly. For the p-space scheme, there are three possible sources of error contributing to the grid dependence of the particle traces: spatial interpolation of the velocity field, temporal integration of the particle paths, and the grid dependence of the flow solution. Since the flow fields in the present and in the previous test case are similar, spatial interpolation and temporal integration errors alone cannot explain the large errors of the p-space scheme which are more than one order of magnitude larger than in the first test case (see Figure 4, parameter zero in the diagrams). Spatial interpolation errors are even smaller in the present test case due to the smaller grid spacing. Because temporal integration errors would show a significant dependence on the integration time step, they do not play a major role here. The errors especially for the p-space scheme are independent of the time step. As a consequence, the main reason for the grid dependence of the particle traces is the grid dependence of the flow solution. In order to determine the grid dependence of the flow field, the velocity was monitored at a position in the interior of the flow region. At this

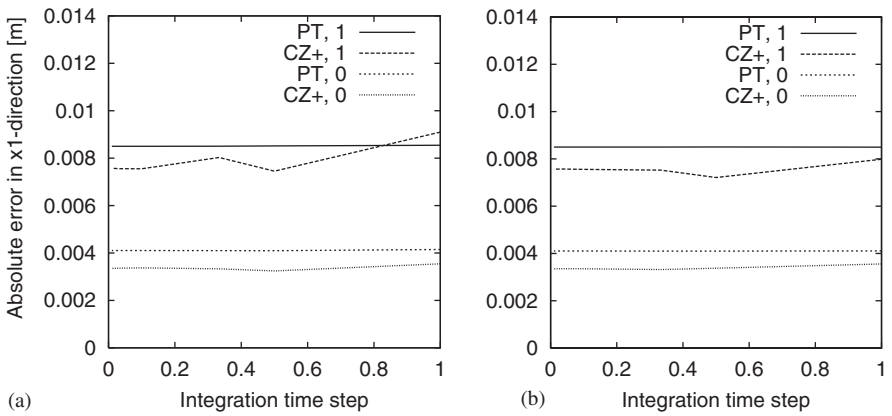


Figure 6. Absolute errors of the c-space scheme CZ+ and the p-space scheme PT for a numerically calculated velocity field. Different degrees of trigonometric grid disturbance (zero corresponds to the orthogonal grid) and the integration schemes RK2 (left) and RK3 (right) were applied. Integration time step in units of a typical simulation time step used in a time-dependent simulation of this flow problem with a CFL number of 2.6.

monitoring position, the velocity on the disturbed coarse grid and the reference grid differs by 0.66 per cent. For the orthogonal coarse grid the velocity difference at the monitoring point is about 0.37 per cent. Obviously, there is a weak grid dependence of the flow solution even in the orthogonal case. Even though such a weak grid dependence (0.37–0.66 per cent) is typically of minor importance for the overall accuracy of the predicted flow field and therefore accepted in practical applications, its effect on the particle traces seems to be larger than the difference between the PT scheme and the CZ+ scheme.

This test case indicates that in practical applications the grid dependence of the flow solution, even if only weakly pronounced, is likely to dominate the overall error of the particle tracing schemes, especially if a flow solver of second-order accuracy in space is applied. Although the p-space scheme PT produces considerably more accurate particle traces than the c-space scheme CZ+ with respect to a given velocity field on a distorted grid, in the case of a numerically calculated velocity field the difference between the schemes with respect to the real particle traces based on the exact solution of the Navier–Stokes equations can become marginal. This suggests that for practical CFD applications the accuracy of CZ+ is equivalent to that of the PT scheme and therefore sufficient.

It is important to note that the situation might be different for the calculation of streambands. Streambands can be calculated by solving a differential equation for the local rotation in addition to the particle's equation of motion [16]. In principle, it is possible to integrate the equation of motion in c-space and the equation of rotation in p-space, thereby avoiding the discontinuity of the c-space velocity in the second integration. However, the impact on the accuracy of streambands calculated with CZ+ compared with PT is an open question. As an alternative to solving a second differential equation, streambands can be constructed by combining two adjacent particle traces with surface elements. In this way only an approximation of the local rotation is obtained, but also additional information about expansion and shear.

5. INVESTIGATION OF PERFORMANCE

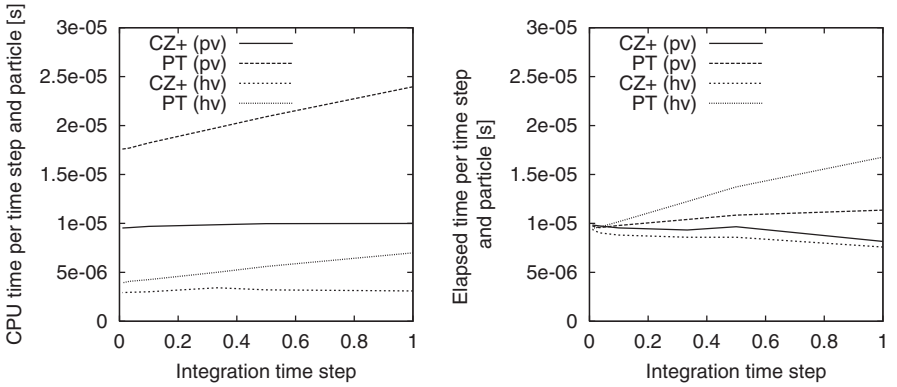
5.1. Test case

Within a realistic multiblock grid and flow field taken from a practical application (stirred vessel simulation), more than 10^5 particles were traced while measuring the time spent for particle tracing. A multiblock grid was chosen to take the computational resources into account which are needed to exchange particles between neighboring blocks. To obtain results which are independent of the parallelization strategy (shared versus distributed memory), these investigations were performed on one processor only. Since the present implementation is fixed to a distributed memory parallelization, and since it is well known that the parallel performance of this method is dominated by the non-controllable load-balancing efficiency, no attempt is made to measure parallel efficiencies. This topic including different parallelization strategies has been addressed in a variety of other papers, e.g., References [17–19].

The performance measurements were carried out on a scalar computer (Sun Ultra 1 workstation, 200 MHz UltraSparc processor) and a vector computer (Fujitsu VPP300). Partly vectorized particle tracing codes more suitable for scalar computers as well as highly vectorized codes were examined. Both the p-space scheme PT and the c-space scheme CZ+ were considered. The integration scheme RK2 was applied to calculate a series of different time steps.

5.2. Results

In Figure 7 the average particle tracing time spent for the advection of one particle over one time step is shown for the vector and the scalar computer. Whereas for CZ+ the particle tracing time is almost independent of the time step size, for PT it is found to be larger for larger time steps. This is due to the iterative point location in the p-space scheme. For large time steps more iterations are necessary to perform a point location of a new particle position starting from the old one.



(a) VPP300 (b) Sun Ultra 1

Figure 7. Performance comparison of the CZ+ and PT schemes for a vector (left) and a scalar computer (right). Both, partly vectorized (pv) and highly vectorized codes (hv) were examined. On the vector computer the function used for time measurement provided the CPU time and on the scalar computer the elapsed time. Integration time step in units of the flow simulation time step.

The acceleration on the vector computer due to code vectorization is different for the two schemes. The highly vectorized code of CZ+ was observed to be about three times faster than the partly vectorized code. For PT the acceleration primarily due to vectorization of the iterative point location is even larger (3–5-fold), since the partly vectorized code is especially inefficient on a vector computer. This is different for CZ+, since its partly vectorized code already achieves a relatively high efficiency compared with PT. The vectorization efficiency can be seen best when turning off the vector unit of the VPP300, thereby using only the scalar unit for the computations. Using the vector unit, the highly vectorized code of CZ+ was found to be about 9–12 times faster than using the scalar unit only, whereas it is only five to six times faster for PT. Obviously, the vectorization efficiency of PT is considerably smaller than for CZ+, which is due to the difficulties with the vectorization of the iterative point location in the p-space scheme (see Section 3.4.2). Furthermore, the comparison between scalar unit and vector unit shows that vectorization of the code is indispensable in order to use the considerably faster vector unit instead of the slow scalar unit.

Whereas on the workstation CZ+ is comparable to (slightly faster than) the partly vectorized code of PT, on the vector computer the highly vectorized code of CZ+ is found to be about 1.3 to more than two times faster than the corresponding code of PT. It is important to note that the result for PT was only achieved owing to much greater algorithmic efforts and additional memory resources required for the vectorization of the iterative point location. Owing to this algorithmic overhead, on the workstation the highly vectorized code of PT is significantly slower than the partly vectorized code. Hence two different codes are necessary to obtain a PT scheme which is optimized for both scalar and vector computers. On the other hand, in the case of CZ+ there is almost no performance difference between the two codes on the workstation, so that the highly vectorized code can be used for both platforms.

6. APPLICATION

The particle tracing scheme CZ+ was applied for the visualization of different flow problems ranging from stirred vessel flows [2] to direct numerical simulations (DNS) of a turbulent channel flow. In the following a DNS application is presented.

At our institute, extensive direct numerical simulations of a turbulent channel flow are carried out, which are of great interest for turbulence research. The application discussed here is based on a computational grid with about seven million control volumes, resulting in more than 200 MB of solution data at every time step. Owing to the limitation of storage space, using conventional post-processing methods a temporally highly resolved visualization of this flow simulation is practically impossible. Applying the co-visualization approach described in Section 2, the data were reduced drastically so that a visualization at high temporal resolution was achieved. In every time step the output of the simulation program was restricted to particle tracing data instead of the flow solution. Two hundred particle start positions were specified at the inlet of the channel, where all 10 time steps new particles were created. In this way 6300 time steps of the simulation were visualized. The complete flow solution would have needed about 1.3 TB of storage space, whereas only 5.5 GB were needed for the particle data. This corresponds to a data reduction factor of about 240.

Timeline and streakline visualizations of the channel flow are shown in Figure 8, demonstrating the high degree of turbulence present in the flow. Additional resources (images and

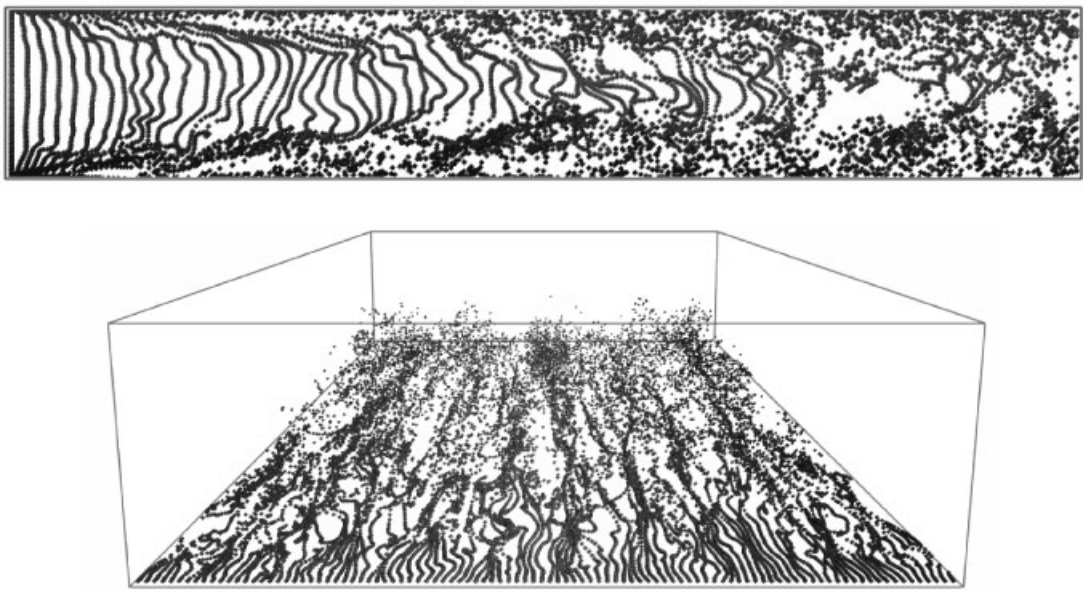


Figure 8. Timeline visualization (top) and streakline visualization (bottom) of the three-dimensional turbulent flow in a channel (direct numerical simulation, $Re = 3300$). The particles are started along a vertical and a horizontal line at the inlet of the channel, respectively.

video sequences) related to this flow problem can be found at the URL http://www.lstm.uni-erlangen.de/SFB603_C3/applications/index_e.html

7. CONCLUSIONS

One p-space and three c-space particle tracing schemes integrated into a parallel multiblock flow simulation program were described. Optimization of the schemes for vector and parallel-vector computers was an important issue. Comparing all c-space schemes investigated, the newly developed, improved CZ+ scheme turned out to be considerably more accurate and more efficient with respect to the operation count. It combines the integration in c-space with trilinear interpolation of the p-space velocity, thereby avoiding the errors introduced by the trilinear interpolation of the c-space velocity.

With respect to an analytically prescribed velocity field, the p-space scheme PT was found to be the most accurate of the schemes applied. However, considering the error introduced by the flow solver by using a numerically calculated velocity field, the accuracy difference between PT and CZ+ was found to be negligible. The error due to a weak grid dependence of the flow solution generally accepted in practical applications dominated the overall error of the particle traces, indicating that for practical CFD applications based on a second-order flow solver the CZ+ scheme is equivalent to PT with respect to accuracy.

Regarding the performance of the particle tracing schemes, on a workstation the CZ+ scheme was found to be comparable to (slightly faster than) the PT scheme, whereas on a

vector computer CZ+ was up to two times faster. Owing to the iterative point location in the p-space scheme, the algorithmic efforts and memory resources required for the vectorization of PT were considerably higher than for CZ+, leading to a smaller vectorization efficiency of PT.

Summarizing these results, with respect to practical CFD applications the c-space scheme CZ+ was found to be comparable to or—with respect to the performance on vector computers—better than the p-space scheme PT. An application of CZ+ to an extensive direct numerical simulation of a turbulent channel flow was presented.

ACKNOWLEDGEMENTS

The present work was carried out as part of the Center of Excellence SFB 603 'Model-Based Analysis and Visualization of Complex Scenes and Sensor Data' at the University of Erlangen–Nuremberg. Financial support by the Deutsche Forschungsgemeinschaft is gratefully acknowledged. We thank René Volkert at LSTM for providing the direct numerical simulation of the turbulent channel flow, which was carried out with financial support of the Stiftung Volkswagenwerk (research project 'Entwicklung statistischer Turbulenzmodelle auf der Basis einer kombiniert experimentell/numerisch erzeugten Datenbasis').

REFERENCES

1. Lane DA. Visualization of numerical unsteady fluid flows. Technical Report NAS-95-017, Numerical Aerospace Simulation Facility (NAS), NASA Ames Research Center, Moffet Field, CA, 1995.
2. Schäfer F, Breuer M. Integrated particle tracing within a parallel multiblock flow simulation program: validation and application. In: B. Girod, H. Niemann, and H.-P. Seidel (eds), *Vision, Modeling and Visualization '99*, pp. 347–356. Infix, Sankt Augustin, Germany; 1999.
3. Shirayama S. Processing of computed vector fields for visualization. *Journal of Computational Physics* 1993; **106**:30–41.
4. Sadarjoen A, van Walsum Th, Hin A, Post FH. Particle tracing algorithms for 3D curvilinear grids. *Fifth Eurographics Workshop on Visualization in Scientific Computing*, 1994.
5. Kenwright DN, Lane DA. Optimization of time-dependent particle tracing using tetrahedral decomposition. In: G. M. Nielson and D. Silver (eds), *Visualization '95*, pp. 321–328. IEEE Computer Society, IEEE Computer Society Press: Los Alamitos, CA; 1995.
6. Globus A. A Software Model for Visualization of Large Unsteady 3-D CFD Results. Technical Report RNR-92-031, NASA Ames Research Center, 1992.
7. Haimes R. pV3: A distributed system for large-scale unsteady CFD visualization. AIAA Paper 94-0321, American Institute of Aeronautics and Astronautics, 1994.
8. Durst F, Schäfer M. A parallel block-structured multigrid method for the prediction of incompressible flows. *International Journal for Numerical Methods in Fluids* 1996; **22**:549–565.
9. Durst F, Schäfer M, Wechsler K. Efficient simulation of incompressible viscous flows on parallel computers. In: *Flow Simulation with High-Performance Computers II*, Vol. 52 of *Notes on Numerical Fluid Mechanics*, pp. 87–101. Vieweg: Braunschweig, 1996.
10. Deuflhard P, Bornemann F. *Numerische Mathematik II*. Walter de Gruyter: Berlin, New York; 1994.
11. Strehmel K, Weiner R. *Linear-implizite Runge-Kutta-Methoden und ihre Anwendung*. Teubner, 1992.
12. Yeung PK, Pope SB. An algorithm for tracking fluid particles in numerical simulations of homogeneous turbulence. *Journal of Computational Physics* 1988; **79**:373–416.
13. Kontomaris K, Hanratty TJ. An algorithm for tracking fluid particles in a spectral simulation of turbulent channel flow. *Journal of Computational Physics* 1992; **103**:231–242.
14. Darmofal DL, Haimes R. An analysis of 3D particle path integration algorithms. *Journal of Computational Physics* 1996; **123**:182–195.
15. Teitzel C, Grosso R, Ertl T. Efficient and reliable integration methods for particle tracing in unsteady flows on discrete meshes. In: W. Lefer and M. Grave (eds), *Visualization in Scientific Computing '97*, pp. 31–41. Springer: Vienna; 1997.
16. Darmofal D, Haimes R. Visualization of 3-D vector fields: variations on a stream. *AIAA Aerospace Sciences Conference*, 1992. AIAA Paper 92-0074.

17. Tsuji Y. Discrete element modelling of clusters in gas-solid Flows. *Symposium on Numerical Methods in Multiphase Flows*, *ASME Fluids Engineering Division Summer Meeting*, Vol. 1, p. 3, San Diego, CA, USA, 7–11 July 1996.
18. Tysinger TL, Missaghi M. A combined shared-memory and distributed-memory model for computation of coupled Lagrangian dispersed phase and Eulerian gas phase combustion. *Proc. of the Int. Conference of Recent Developments and Advances Using Parallel Computers*, *Parallel CFD '97*, Manchester, U.K., 19–21 May 1997.
19. Wassen E, Frank Th, Yu Q. A comparison of parallel algorithms for the numerical simulation of multiphase flows. *Proc. of the 1. Euro-Conference on Parallel and Distributed Computing for Computational Mechanics*, Lochinver, Scotland, U.K., 26 April–1 May 1997.

