A SIMPLE EULERIAN FINITE-VOLUME METHOD FOR COMPRESSIBLE FLUIDS IN DOMAINS WITH MOVING BOUNDARIES

ALINA CHERTOCK * AND ALEXANDER KURGANOV †

Abstract. We introduce a new simple Eulerian method for treatment of moving boundaries in compressible fluid computations. Our approach is based on the extension of the interface tracking method recently introduced in the context of multifluids. The fluid domain is placed in a rectangular computational domain of a fixed size, which is divided into Cartesian cells. At every time moment, there are three types of cells: internal, boundary, and external ones. The numerical solution is evolved in internal cells only. The numerical fluxes at the cells near the boundary are computed using the technique from [A. Chertock, S. Karni and A. Kurganov *M2AN Math. Model. Numer. Anal.*, submitted] combined with a solid wall ghost-cell extrapolation and an interpolation in the phase space. The proposed computational framework is general and may be used in conjunction with one's favorite finite-volume method. The robustness of the new approach is illustrated on a number of one- and two-dimensional numerical examples.

Key words. Compressible Euler equations, ghost-cell extrapolation, moving boundaries, semi-discrete finite-volume schemes.

AMS subject classifications. 65M99, 35L65, 76N99

1. Introduction

We are interested in developing a simple, accurate, and robust numerical method for computing compressible fluids in the domains with moving boundaries. In the two-dimensional (2-D) case, the governing equations are the compressible Euler equations:

$$\begin{pmatrix}\rho\\ \rho u\\ \rho v\\ E\end{pmatrix}_t+\begin{pmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ u(E+p)\end{pmatrix}_x+\begin{pmatrix}\rho v\\ \rho uv\\ \rho v^2+p\\ v(E+p)\end{pmatrix}_y=0, \tag{1.1}$$

where ρ is the fluid density, u and v are the velocities, E is the total energy, and p is the pressure. The system (1.1) is closed using the equation of state (EOS), which, for ideal gases, reads:

$$E=\frac{p}{\gamma-1}+\frac{\rho}{2}(u^2+v^2),\quad \gamma=\text{const}.$$

We also introduce the notation $c:=\sqrt{\gamma p/\rho}$ for the speed of sound, which will be used throughout the paper.

Our goal is to apply Eulerian finite-volume (FV) methods to the problems with changing geometries. To this end, we place the fluid domain into the computational domain of a fixed size, which is divided into Cartesian cells. At every time moment, each cell is marked as either internal, external, or boundary one. The internal cells are fully occupied by the gas, the external cells are located outside of the fluid domain

*Department of Mathematics, North Carolina State University, Raleigh, NC 27695, (chertock@math.ncsu.edu).

†Mathematics Department, Tulane University, New Orleans, LA 70118, (kurganov@math.tulane.edu).

and play the role of the so-called ghost cells, while the boundary cells form a thin layer between the internal and external ones. The boundary cells have to be introduced since in the case of moving boundaries, the fluid domain boundary cannot, in general, be forced to coincide with the cell edges. As a result, the boundary cells are only partially filled with the gas, which is very inconvenient since within the FV computational framework, numerical solutions are represented in terms of the cell averages. One of the possible ways to treat the boundary cells is to split each of them into two smaller cells: the internal and the external ones. However, this would significantly increase the complexity of the entire solution algorithm and may lead to very small time steps (see, e.g., [1, 4, 11]. We prefer an alternative, simpler approach, in which the averages are computed over the internal cells only and the data contained in the boundary cells are not used for the computation of numerical fluxes. We treat the boundary cells similarly to the way the so-called "mixed" cells have been treated in [5]. Namely, we only approximate the point values at the edges of these cells, required in the numerical flux computations. These point values are obtained using the solid wall extrapolation followed by the interpolation in the phase space (by solving the Riemann problem between the internal cell averages and the extrapolated ones). The numerical solution is then evolved in internal cells only using a FV method.

The proposed computational framework is general and may be used in conjunction with one's favorite FV method. We refer the reader to [9, 12, 17, 22], where the description of a variety of modern high-order FV methods can be found. In this paper, we have used the semi-discrete second-order central-upwind scheme developed in [13, 14, 15]. This Godunov-type scheme enjoys all major advantages of Riemann-problem-solver-free, non-oscillatory central schemes and, at the same time, have a certain "built-in" upwind nature.

The paper is organized as follows. The description of the new one-dimensional (1-D) and 2-D methods are presented in §2 and §3, respectively. The key part of this paper — a special Eulerian way of treating moving boundaries in one and two space dimensions — can be found in §2.1 and §3.1. Finally, in §5, we demonstrate the high resolution and robustness of the new schemes in a series of 1-D and 2-D numerical experiments.

2. One-Dimensional Method

We begin with a description of a general 1-D semi-discrete FV framework. The boundary cells treatment is presented in §2.1.

We consider the 1-D version of the system (1.1):

$$\mathbf{U}_t+\mathbf{f}(\mathbf{U})_x=0, \tag{2.1}$$

where $\mathbf{U}:=(\rho,\rho u,E)^T$ and $\mathbf{f}(\mathbf{U}):=(\rho u,\rho u^2+p,u(E+p))^T$. For simplicity, we divide the computational domain into FV cells $C_j:=[x_{j-\frac12},x_{j+\frac12}]$ of a uniform size Δx. A semi-discrete FV scheme for (2.1) is the following system of ODEs:

$$\frac{d}{dt}\overline{\mathbf{U}}_j(t)=-\frac{\mathbf{H}_{j+\frac12}(t)-\mathbf{H}_{j-\frac12}(t)}{\Delta x}, \tag{2.2}$$

where $\overline{\mathbf{U}}_j(t)$ are approximations of the cell averages of the solution:

$$\overline{\mathbf{U}}_j(t)\approx\frac{1}{\Delta x}\int\limits_{C_j}\mathbf{U}(x,t)\,dx,$$

and $\mathbf{H}_{j+\frac12}(t)=\mathcal{H}(\mathbf{U}^-_{j+\frac12}(t),\mathbf{U}^+_{j+\frac12}(t))$ is a numerical flux function. Here,

$$\mathbf{U}^+_{j+\frac12}(t):=\mathcal{P}_{j+1}(x_{j+\frac12},t)\quad\text{and}\quad\mathbf{U}^-_{j+\frac12}(t):=\mathcal{P}_j(x_{j+\frac12},t) \tag{2.3}$$

are the right and the left values at $x=x_{j\pm\frac12}$ of a conservative, (essentially) non-oscillatory, piecewise polynomial interpolant

$$\widetilde{\mathbf{U}}(x,t)=\mathcal{P}_j(x,t),\qquad x_{j-\frac12}<x<x_{j+\frac12}, \tag{2.4}$$

which is reconstructed at each time step from the previously computes cell averages, $\{\overline{\mathbf{U}}_j(t)\}$. Notice that the formal spatial order of accuracy of the resulting scheme depends on the order of the piecewise polynomial reconstruction (2.4).

To complete the description of a concrete FV method the following three items are to be specified (selected):

(*i*) Numerical flux. A variety of reliable functions $\mathcal{H}$ is available in the literature (see, e.g., [9, 12, 17, 22], and references therein). In all our numerical experiments, we have used the central-upwind flux [13, 14]:

$$\mathbf{H}_{j+\frac12}(t)=\frac{a^+_{j+\frac12}\mathbf{f}(\mathbf{U}^-_{j+\frac12})-a^-_{j+\frac12}\mathbf{f}(\mathbf{U}^+_{j+\frac12},)}{a^+_{j+\frac12}-a^-_{j+\frac12}}+\frac{a^+_{j+\frac12}a^-_{j+\frac12}}{a^+_{j+\frac12}-a^-_{j+\frac12}}\Big[\mathbf{U}^+_{j+\frac12}-\mathbf{U}^-_{j+\frac12}\Big],$$

where $a^\pm_{j+\frac12}$ are the right- and left-sided local speeds obtained from the largest and the smallest eigenvalues of the Jacobian $\frac{\partial\mathbf{F}}{\partial\mathbf{U}}$:

$$\begin{aligned}a^+_{j+\frac12}&=\max\Big\{u^+_{j+\frac12}+c^+_{j+\frac12},u^-_{j+\frac12}+c^-_{j+\frac12},0\Big\},\\ a^-_{j+\frac12}&=\min\Big\{u^+_{j+\frac12}-c^+_{j+\frac12},u^-_{j+\frac12}-c^-_{j+\frac12},0\Big\}.\end{aligned}$$

Note that the quantities $\overline{\mathbf{U}}_j$, $\mathbf{U}^\pm_{j+\frac12}$, $(\mathbf{U}_x)_j$, $p^\pm_{j+\frac12}$, $c^\pm_{j+\frac12}$, and $a^\pm_{j+\frac12}$ depend on t, but from now on we, for simplicity, will suppress this dependence in our notation;

(*ii*) Reconstruction. In this paper, we will restrict our consideration to second-order schemes that are typically based on piecewise linear reconstructions, which can be written as:

$$\widetilde{\mathbf{U}}(x):=\overline{\mathbf{U}}_j+(\mathbf{U}_x)_j(x-x_j),\quad x_{j-\frac12}<x<x_{j+\frac12}. \tag{2.5}$$

The numerical derivatives $(\mathbf{U}_x)_j$ are (at least) first-order, componentwise approximations of $\mathbf{U}_x(x_j,t)$, computed using a nonlinear limiter needed to ensure a non-oscillatory nature of the reconstruction (2.5). A library of such limiters is available (see, e.g., [9, 12, 16, 17, 18, 19, 20]), and one can compute the numerical derivatives using one's favorite limiter. In our numerical experiments, we have used the generalized minmod limiter [16, 18, 19, 20]:

$$(\mathbf{U}_x)_j=\text{minmod}\left(\theta\frac{\overline{\mathbf{U}}_j-\overline{\mathbf{U}}_{j-1}}{\Delta x},\frac{\overline{\mathbf{U}}_{j+1}-\overline{\mathbf{U}}_{j-1}}{2\Delta x},\theta\frac{\overline{\mathbf{U}}_{j+1}-\overline{\mathbf{U}}_j}{\Delta x}\right),\quad\theta\in[1,2], \tag{2.6}$$

where the minmod function is defined as:

$$\text{minmod}(z_1,z_2,...):=\begin{cases}\min_j\{z_j\}, & \text{if }\ z_j>0\ \ \forall j,\\ \max_j\{z_j\}, & \text{if }\ z_j<0\ \ \forall j,\\ 0, & \text{otherwise,}\end{cases} \tag{2.7}$$

and the parameter θ can be used to control the amount of numerical viscosity present in the resulting scheme. Let us recall that larger values of θ correspond to less dissipative but, in general, more oscillatory reconstructions.

(*iii*) ODE solver. The resulting semi-discretization (2.2) is a system of time-dependent ODEs, which should be solved by a stable ODE solver of an appropriate order.

2.1. Boundary Cell Treatment in 1-D. Let us assume that our computational domain, which is sufficiently large to contain the entire fluid domain at all times, is divided into N uniform cells $\{C_j;\ j=1,\dots,N\}$ and that the right boundary of the fluid domain is a moving solid wall with a given equation of motion. We also assume that at some time $t\geq 0$, the right boundary $x=x_{\rm B}(t)\leq x_N$ is contained in cell J, that is, $x_{J-\frac{1}{2}}\leq x_{\rm B}(t)\leq x_{J+\frac{1}{2}}$, and the computed solution, realized by the cell averages of the conserved quantities, $\{\overline{\mathbf{U}}_j(t);\ j=1,\dots,J\}$, is available.

We now have to evolve the computed solution from time level t to the new time level $t+\Delta t$ using a semi-discrete FV scheme, described in the beginning of §2. One step of the proposed time evolution algorithm consists of the following stages.

We first evolve the "internal" part of the solution, $\{\overline{\mathbf{U}}_j(t);\ j=1,\dots,J-1\}$. According to (2.2), this means that we need to compute the following set of numerical fluxes: $\{\mathbf{H}_{j+\frac{1}{2}}(t);\ j=0,\dots,J-1\}$, which are obtained using the corresponding point values: $\{\mathbf{U}^{\pm}_{j+\frac{1}{2}}(t);\ j=0,\dots,J-1\}$, calculated by a piecewise polynomial (say, for simplicity, piecewise linear) reconstruction (2.3)–(2.4). Such a reconstruction employs a nonlinear limiter (the minmod (2.6)–(2.7) or an alternative one) and therefore, in order to obtain a polynomial piece in any given cell, we need (at least) two neighboring cell averages available. This way, we will have the reconstructed pieces $\{\mathcal{P}_j\}$ for all j except for $j=J-1$ and $j=J$ since we refrain from using the cell averages at the boundary cell J and since no data is available at the external cells $\{C_j;\ j=J+1,\dots,N\}$. We thus need a special procedure for the computation of the following three point values: $\mathbf{U}^{+}_{J-\frac{3}{2}}$, $\mathbf{U}^{-}_{J-\frac{1}{2}}$, and $\mathbf{U}^{+}_{J-\frac{1}{2}}$.

To obtain $\mathbf{U}^{+}_{J-\frac{1}{2}}$, we use our assumption that the moving boundary, present in cell J, is a solid wall. Therefore, we consider the solid wall extrapolation of the solution values in cell $(J-1)$,

$$\overline{\rho}_{J-1},\quad u_{J-1}=\frac{\overline{(\rho u)}_{J-1}}{\overline{\rho}_{J-1}},\quad p_{J-1}=(\gamma-1)\left[\overline{E}_{J-1}-\frac{\overline{\rho}_{J-1}u^2_{J-1}}{2}\right],$$

to obtain the following ghost-cell values:

$$\rho_{\rm gh}:=\overline{\rho}_{J-1},\quad u_{\rm gh}:=2\dot{x}_{\rm B}(t)-u_{J-1},\quad p_{\rm gh}:=p_{J-1},$$

where $\dot{x}_{\rm B}(t)$ is the velocity of the wall at time t. The required point values $\mathbf{U}^{+}_{J-\frac{1}{2}}$ are then computed using the interpolation between $(\overline{\rho}_{J-1},u_{J-1},p_{J-1})$ and $(\rho_{\rm gh},u_{\rm gh},p_{\rm gh})$, which is, following the idea of the multi-fluid algorithms in [6] and [5], performed in the phase space. Namely, we exactly solve the Riemann problem between the above two states. Since the density and the pressure values on the left and on the right are the same, the solution of the Riemann problem has a very simple structure: it consists of either two shocks or two rarefaction waves. In particular, if $u_{\rm gh}<u_{J-1}$, the solution consists of three constant states, connected by two shock waves (see Figure 2.1). The intermediate state (ρ_*,u_*,p_*), which is easy to compute in this case (consult, e.g.,

[22]), is equal to:

$$p_*=p_{J-1}+\frac{(\gamma+1)\overline{\rho}_{J-1}\big(\dot{x}_{\rm B}(t)-u_{J-1}\big)^2}{4}\left[1+\sqrt{1+\left(\frac{4c_{J-1}}{(\gamma+1)\big(\dot{x}_{\rm B}(t)-u_{J-1}\big)}\right)^2}\right],$$

(2.8)

$$u_*=\dot{x}_{\rm B}(t),\qquad \rho_*=\overline{\rho}_{J-1}\frac{(\gamma-1)p_{J-1}+(\gamma+1)p_*}{(\gamma+1)p_{J-1}+(\gamma-1)p_*}.$$

If $u_{\rm gh}>u_{J-1}$, the solution consists of three constant states, connected by two rarefaction waves, and the intermediate state (ρ_*,u_*,p_*) is given by (once again, consult, e.g., [22]):

$$p_*=p_{J-1}\left(1-\frac{(\gamma-1)\big(\dot{x}_{\rm B}(t)-u_{J-1}\big)}{2c_{J-1}}\right)^{2\gamma/(\gamma-1)},\quad u_*=\dot{x}_{\rm B}(t),\quad \rho_*=\overline{\rho}_{J-1}\left(\frac{p_*}{p_{J-1}}\right)^{1/\gamma}.$$

(2.9)

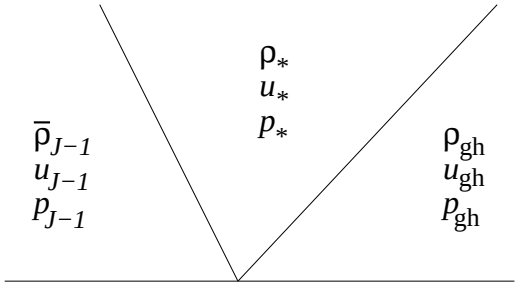

FIG. 2.1. A shock-shock self-similar solution of the Riemann problem between $(\overline{\rho}_{J-1},u_{J-1},p_{J-1})$ and $(\rho_{\rm gh},u_{\rm gh},p_{\rm gh})$, presented in the (x,t)-plane.

Once the intermediate state of the Riemann problem solution is available, we compute the required values at the left endpoint of cell J according to the following algorithm:

```
if u_* > 0  then
```

$$\mathbf{U}^+_{J-\frac12}=\begin{cases}\mathbf{U}_*, & \text{if } u_*-c_*<0,\\ \overline{\mathbf{U}}_{J-1}, & \text{otherwise,}\end{cases}$$

```
else
```

$$\mathbf{U}^+_{J-\frac12}=\begin{cases}\mathbf{U}_*, & \text{if } u_*+c_*>0,\\ \overline{\mathbf{U}}_{\rm gh}, & \text{otherwise,}\end{cases}$$

where

$$\mathbf{U}_*:=\left(\rho_*,\ \rho_*u_*,\ \frac{p_*}{\gamma-1}+\frac{\rho_*}{2}(u_*)^2\right)^T,\quad \overline{\mathbf{U}}_{\rm gh}:=\left(\rho_{\rm gh},\ \rho_{\rm gh}u_{\rm gh},\ \frac{p_{\rm gh}}{\gamma-1}+\frac{\rho_{\rm gh}}{2}(u_{\rm gh})^2\right)^T.$$

The point values $\mathbf{U}^+_{J-\frac12}$ are then used to reconstruct a conservative linear piece (2.5) at the neighboring cell C_{J-1}. The reconstruction is made non-oscillatory by computing

the derivatives $(\mathbf{U}_x)_{J-1}$ with the help of the minmod limiter, applied to the cell averages at cell $(J-1)$ and to the corresponding point values, already computed at the endpoints of the cells C_{J-2} and C_J:

$$(\mathbf{U}_x)_{J-1} = \text{minmod}\left(\frac{\overline{\mathbf{U}}_{J-1}-\mathbf{U}^-_{J-\frac{3}{2}}}{\Delta x/2}, \frac{\mathbf{U}^+_{J-\frac{1}{2}}-\overline{\mathbf{U}}_{J-1}}{\Delta x/2}\right), \tag{2.10}$$

where the minmod function, given by (2.7), is applied componentwise, see Figure 2.2.

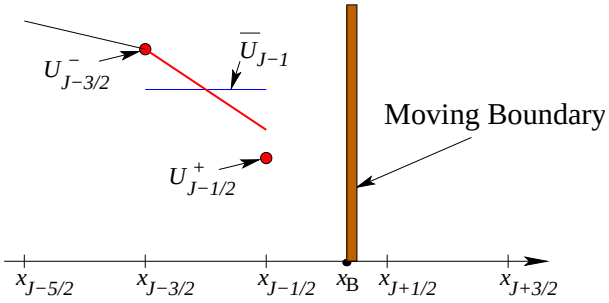

FIG. 2.2. Special minmod reconstruction in cell $(J-1)$.

With the values $\{\mathbf{U}^{\pm}_{j+\frac{1}{2}};\ j=0,\ldots,J-1\}$ at hand, the cell averages $\{\overline{\mathbf{U}}_j\}$ are evolved from time t to time $t+\Delta t$ according to the FV scheme (2.2). After this, the location of the moving wall is updated, and three scenarios are possible: the right boundary either remains in cell J or crosses over to one of its neighboring cells $(J\pm1)$ (due to the CFL condition, the boundary may not move by more than Δx per one time step).

• If $x_B(t+\Delta t)$ remains in cell J, the evolution step is completed and we proceed to the next time step.

• If $x_B(t+\Delta t)$ is in cell $(J-1)$, then this cell becomes the new boundary cell, while cell J turns into an exterior one and the computed values $\overline{\mathbf{U}}_J(t+\Delta t)$ will not be used by our method any more.

• If the boundary moves to cell $(J+1)$, which becomes a new boundary cell, while cell J turns into an interior one, we follow the approach from [5, 6] and once again use the solution of the Riemann problem between $(\overline{\rho}_{J-1}, u_{J-1}, p_{J-1})$ and $(\rho_{gh}, u_{gh}, p_{gh})$ to obtain $\overline{\mathbf{U}}_J(t+\Delta t)$. The correction procedure is summarized in the following algorithm:

$$\texttt{if } x_B(t+\Delta t)\in C_{J+1} \texttt{ then set } \overline{\mathbf{U}}_J(t+\Delta t) := \mathbf{U}_*$$

This completes one evolution step of the proposed 1-D Eulerian method for the case of moving right boundary. The case of moving left boundary is treated similarly.

3. Two-Dimensional Method

As in §2, we begin with a description of a general 2-D semi-discrete FV scheme, while the presentation of the boundary cells treatment is postponed to §3.1.

We denote by $C_{j,k}$ the computational cells $C_{j,k}:=[x_{j-\frac12},x_{j+\frac12}]\times[y_{k-\frac12},y_{k+\frac12}]$ of a uniform size $\Delta x\Delta y$. As in the 1-D case, the cell averages,

$$\overline{\mathbf{U}}_{j,k}(t)\approx\frac{1}{\Delta x\Delta y}\iint\limits_{C_{j,k}}\mathbf{U}(x,y,t)\,dx\,dy,\quad \mathbf{U}:=(\rho,\rho u,\rho v,E)^T,$$

are evolved in time according to the semi-discrete FV scheme:

$$\frac{d}{dt}\overline{\mathbf{U}}_{j,k}(t)=-\frac{\mathbf{H}^x_{j+\frac12,k}(t)-\mathbf{H}^x_{j-\frac12,k}(t)}{\Delta x}-\frac{\mathbf{H}^y_{j,k+\frac12}(t)-\mathbf{H}^y_{j,k-\frac12}(t)}{\Delta y},\tag{3.1}$$

where $\mathbf{H}^x_{j+\frac12,k}(t)$ and $\mathbf{H}^y_{j,k+\frac12}(t)$ are numerical fluxes, which are to be computed with the help of a conservative, (essentially) non-oscillatory piecewise polynomial reconstruction.

As in the 1-D case, the description of the scheme will be completed by selecting specific numerical flux functions, a piecewise polynomial reconstruction, and an ODE solver. Once again, we would like to emphasize that our method is not tied to a particular choice of any of these three ingredients. The numerical results presented in §5.2 have been obtained using the following:

(*i*) Numerical fluxes. We have used the second-order central-upwind fluxes ([13, 14]):

$$\mathbf{H}^x_{j+\frac12,k}=\frac{a^+_{j+\frac12,k}\mathbf{f}(\mathbf{U}^{\rm E}_{j,k})-a^-_{j+\frac12,k}\mathbf{f}(\mathbf{U}^{\rm W}_{j+1,k})}{a^+_{j+\frac12,k}-a^-_{j+\frac12,k}}+\frac{a^+_{j+\frac12,k}a^-_{j+\frac12,k}}{a^+_{j+\frac12,k}-a^-_{j+\frac12,k}}\left[\mathbf{U}^{\rm W}_{j+1,k}-\mathbf{U}^{\rm E}_{j,k}\right],$$

$$\mathbf{H}^y_{j,k+\frac12}=\frac{b^+_{j,k+\frac12}\mathbf{g}(\mathbf{U}^{\rm N}_{j,k})-b^-_{j,k+\frac12}\mathbf{g}(\mathbf{U}^{\rm S}_{j,k+1})}{b^+_{j,k+\frac12}-b^-_{j,k+\frac12}}+\frac{b^+_{j,k+\frac12}b^-_{j,k+\frac12}}{b^+_{j,k+\frac12}-b^-_{j,k+\frac12}}\left[\mathbf{U}^{\rm S}_{j,k+1}-\mathbf{U}^{\rm N}_{j,k}\right],$$

where $\mathbf{f}(\mathbf{U}):=(\rho u,\rho u^2+p,\rho uv,u(E+p))^T$, $\mathbf{g}(\mathbf{U}):=(\rho u,\rho uv,\rho v^2+p,v(E+p))^T$, $\mathbf{U}^{\rm E,W,N,S}_{j,k}$ are the point values of a piecewise linear reconstruction $\widetilde{\mathbf{U}}$, and the local one-sided speeds of propagation are computed by:

$$\begin{aligned}
a^+_{j+\frac12,k}&=\max\left\{u^{\rm E}_{j,k}+c^{\rm E}_{j,k},u^{\rm W}_{j+1,k}+c^{\rm W}_{j+1,k},0\right\},\\
a^-_{j+\frac12,k}&=\min\left\{u^{\rm E}_{j,k}-c^{\rm E}_{j,k},u^{\rm W}_{j+1,k}-c^{\rm W}_{j+1,k},0\right\},\\
b^+_{j,k+\frac12}&=\max\left\{v^{\rm N}_{j,k}+c^{\rm N}_{j,k},v^{\rm S}_{j,k+1}+c^{\rm S}_{j,k+1},0\right\},\\
b^-_{j,k+\frac12}&=\min\left\{v^{\rm N}_{j,k}-c^{\rm N}_{j,k},v^{\rm S}_{j,k+1}-c^{\rm S}_{j,k+1},0\right\}.
\end{aligned}$$

As before, we suppress the dependence of $\overline{\mathbf{U}}_{j,k}$, $\mathbf{U}^{\rm E,W,N,S}_{j,k}$, $p^{\rm E,W,N,S}_{j,k}$, $c^{\rm E,W,N,S}_{j,k}$, $a^{\pm}_{j+\frac12,k}$, and $b^{\pm}_{j,k+\frac12}$ on t to simplify the notation.

(*ii*) Reconstruction. We have utilized the generalized minmod piecewise linear reconstruction:

$$\widetilde{\mathbf{U}}(x,y):=\overline{\mathbf{U}}_{j,k}+(\mathbf{U}_x)_{j,k}(x-x_j)+(\mathbf{U}_y)_{j,k}(y-y_k),\quad (x,y)\in C_{j,k}\tag{3.2}$$

with the corresponding point values at the midpoints of the cell edges:

$$\mathbf{U}^{\rm E(W)}_{j,k}:=\overline{\mathbf{U}}_{j,k}\pm\frac{\Delta x}{2}(\mathbf{U}_x)_{j,k},\qquad \mathbf{U}^{\rm N(S)}_{j,k}:=\overline{\mathbf{U}}_{j,k}\pm\frac{\Delta y}{2}(\mathbf{U}_y)_{j,k}.\tag{3.3}$$

The numerical derivatives $(\mathbf{U}_x)_{j,k}$ and $(\mathbf{U}_y)_{j,k}$ are computed by

$$
\begin{aligned}
(\mathbf{U}_x)_{j,k} &= \text{minmod}\left(\theta\frac{\overline{\mathbf{U}}_{j,k}-\overline{\mathbf{U}}_{j-1,k}}{\Delta x}, \frac{\overline{\mathbf{U}}_{j+1,k}-\overline{\mathbf{U}}_{j-1,k}}{2\Delta x}, \theta\frac{\overline{\mathbf{U}}_{j+1,k}-\overline{\mathbf{U}}_{j,k}}{\Delta x}\right),\\
(\mathbf{U}_y)_{j,k} &= \text{minmod}\left(\theta\frac{\overline{\mathbf{U}}_{j,k}-\overline{\mathbf{U}}_{j,k-1}}{\Delta y}, \frac{\overline{\mathbf{U}}_{j,k+1}-\overline{\mathbf{U}}_{j,k-1}}{2\Delta y}, \theta\frac{\overline{\mathbf{U}}_{j,k+1}-\overline{\mathbf{U}}_{j,k}}{\Delta y}\right),
\end{aligned}
\tag{3.4}
$$

where, as in the 1-D case, the parameter $\theta\in[1,2]$.

(*iii*) ODE solver. The system (3.1) is to be solved by a stable ODE solver of an appropriate order.

3.1. Boundary Cell Treatment in 2-D. The extension of the 1-D interface treatment algorithm to 2-D is carried out in an essentially dimension-by-dimension manner.

We assume that at a certain time level $t\geq 0$, the computed solution is available and that cell (J,K) is a boundary cell, while its left neighbor $(J-1,K)$ is an internal cell (see Figure 3.1 (a)). Other possible "near boundary" configurations, shown in Figures 3.1 (b)–(d), are treated similarly.

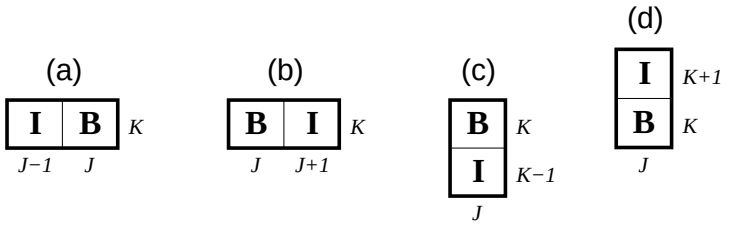

FIG. 3.1. Typical cell configurations near the boundary cell (J,K). The internal cells are denoted by **I**.

Let us first describe how to compute the point values $\mathbf{U}^{\mathrm{W}}_{J,K}$, needed to evaluate the numerical fluxes $\mathbf{H}^x_{J-\frac{1}{2},K}$ in the situation corresponding to Figure 3.1 (a). To this end, we extend the 1-D algorithm described in §2.1 to two space dimensions. The extension is carried out straightforwardly, namely, we solve the Riemann problem in the x-direction between the state in cell $(J-1,K)$,

$$
\begin{aligned}
&\overline{\rho}_{J-1,K},\quad u_{J-1,K}=\frac{\overline{(\rho u)}_{J-1,K}}{\overline{\rho}_{J-1,K}},\quad v_{J-1,K}=\frac{\overline{(\rho v)}_{J-1,K}}{\overline{\rho}_{J-1,K}},\\
&p_{J-1,K}=(\gamma-1)\left[\overline{E}_{J-1,K}-\frac{\overline{\rho}_{J-1,K}}{2}\left(u^2_{J-1,K}+v^2_{J-1,K}\right)\right],
\end{aligned}
$$

and the extrapolated ghost-cell state,

$$
\rho_{\mathrm{gh}}=\overline{\rho}_{J-1,K},\quad u_{\mathrm{gh}}=2u_{\mathrm{B}}(t)-u_{J-1,K},\quad v_{\mathrm{gh}}=v_{\mathrm{B}}(t),\quad p_{\mathrm{gh}}=p_{J-1,K},
$$

where $u_{\mathrm{B}}(t)$ and $v_{\mathrm{B}}(t)$ are the x- and y-velocities of the piece of the boundary, which is located in cell (J,K) at time t. We would like to stress that since we are dealing with a solid boundary with a given equation of motion, the location of the entire boundary as well as its velocities are assumed to be known at every time moment.

The solution of this 1-D Riemann problem can still be easily obtained analytically. If $u_{\rm gh} < u_{J-1,K}$, the solution consists of four constant states, connected by two shocks and a contact wave (see Figure 3.2). The intermediate state values are:

$$p_*^{\rm W} = p_{J-1,K} + \frac{(\gamma+1)\overline{\rho}_{J-1,K}\,(u_{\rm B}(t)-u_{J-1,K})^2}{4}\left[1+\sqrt{1+\left(\frac{4c_{J-1,K}}{(\gamma+1)\,(u_{\rm B}(t)-u_{J-1,K})}\right)^2}\right],$$

$$u_* = u_{\rm B}(t), \qquad \rho_*^{\rm W} = \overline{\rho}_{J-1,K}\frac{(\gamma-1)p_{J-1,K}+(\gamma+1)p_*^{\rm W}}{(\gamma+1)p_{J-1,K}+(\gamma-1)p_*^{\rm W}}.$$

If $u_{\rm gh} > u_{J-1,K}$, the solution consists of four constant states, connected by two rarefactions and a contact wave, with the following intermediate state values:

$$p_*^{\rm W} = p_{J-1,K}\left(1-\frac{(\gamma-1)\,(u_{\rm B}(t)-u_{J-1,K})}{2c_{J-1,K}}\right)^{2\gamma/(\gamma-1)},$$

$$u_* = u_{\rm B}(t), \qquad \rho_*^{\rm W} = \overline{\rho}_{J-1,K}\left(\frac{p_*^{\rm W}}{p_{J-1,K}}\right)^{1/\gamma}.$$

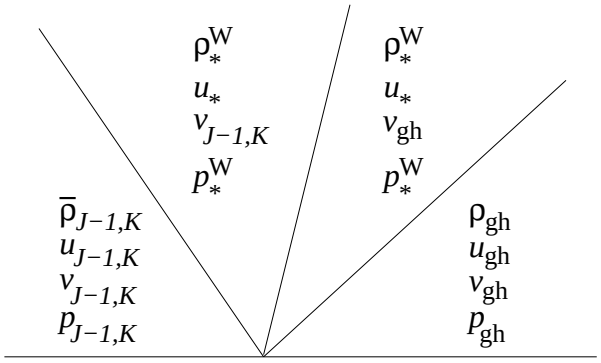

FIG. 3.2. A shock-contact-shock self-similar solution of the Riemann problem in the x-direction between $(\overline{\rho}_{J-1,K}, u_{J-1,K}, v_{J-1,K}, p_{J-1,K})$ and $(\rho_{\rm gh}, u_{\rm gh}, v_{\rm gh}, p_{\rm gh})$, presented in the (x,t)-plane.

Once the intermediate states are available, we obtain the required values at the

midpoint of the left edge of cell (J,K):

$$\texttt{if } u_*>0 \texttt{ then}$$

$$\mathbf{U}_*^{\mathrm{W}} := \left(\rho_*^{\mathrm{W}},\ \rho_*^{\mathrm{W}}u_*,\ \rho_*^{\mathrm{W}}v_{J-1,K},\ \frac{p_*^{\mathrm{W}}}{\gamma-1}+\frac{\rho_*^{\mathrm{W}}}{2}\left[(u_*)^2+(v_{J-1,K})^2\right]\right)^T,$$

$$\mathbf{U}_{J,K}^{\mathrm{W}} = \begin{cases} \mathbf{U}_*^{\mathrm{W}}, & \text{if } \ u_*-c_*^{\mathrm{W}}<0, \\ \overline{\mathbf{U}}_{J-1,K}, & \text{otherwise,} \end{cases}$$

$$\texttt{else}$$

$$\mathbf{U}_*^{\mathrm{W}} := \left(\rho_*^{\mathrm{W}},\ \rho_*^{\mathrm{W}}u_*,\ \rho_*^{\mathrm{W}}v_{\mathrm{gh}},\ \frac{p_*^{\mathrm{W}}}{\gamma-1}+\frac{\rho_*^{\mathrm{W}}}{2}\left[(u_*)^2+(v_{\mathrm{gh}})^2\right]\right)^T,$$

$$\overline{\mathbf{U}}_{\mathrm{gh}} = \left(\rho_{\mathrm{gh}},\ \rho_{\mathrm{gh}}u_{\mathrm{gh}},\ \rho_{\mathrm{gh}}v_{\mathrm{gh}},\ \frac{p_{\mathrm{gh}}}{\gamma-1}+\frac{\rho_{\mathrm{gh}}}{2}\left[(u_{\mathrm{gh}})^2+(v_{\mathrm{gh}})^2\right]\right)^T,$$

$$\mathbf{U}_{J,K}^{\mathrm{W}} = \begin{cases} \mathbf{U}_*^{\mathrm{W}}, & \text{if } \ u_*+c_*^{\mathrm{W}}>0, \\ \overline{\mathbf{U}}_{\mathrm{gh}}, & \text{otherwise.} \end{cases}$$

As in the 1-D case, the point values $\mathbf{U}_{J,K}^{\mathrm{W}}$ are then used to reconstruct a conservative linear piece (3.2) at the neighboring cell $C_{J-1,K}$. This is done similarly to (2.10), and the details are left to the reader.

In the situation corresponding to Figure 3.1 (b), we proceed similarly. Namely, we solve the Riemann problem in the x-direction between the state in cell $(J+1,K)$,

$$\overline{\rho}_{J+1,K}, \quad u_{J+1,K}=\frac{\overline{(\rho u)}_{J+1,K}}{\overline{\rho}_{J+1,K}}, \quad v_{J+1,K}=\frac{\overline{(\rho v)}_{J+1,K}}{\overline{\rho}_{J+1,K}},$$

$$p_{J+1,K}=(\gamma-1)\left[\overline{E}_{J+1,K}-\frac{\overline{\rho}_{J+1,K}}{2}\left(u_{J+1,K}^2+v_{J+1,K}^2\right)\right],$$

and the extrapolated ghost-cell state,

$$\rho_{\mathrm{gh}}=\overline{\rho}_{J+1,K}, \quad u_{\mathrm{gh}}=2u_{\mathrm{B}}(t)-u_{J+1,K}, \quad v_{\mathrm{gh}}=v_{\mathrm{B}}(t), \quad p_{\mathrm{gh}}=p_{J+1,K}.$$

The resulting algorithm for computing $\mathbf{U}^{\rm E}_{J,K}$ is summarized below:

$$
\begin{aligned}
&\texttt{if } u_*<0 \texttt{ then}\\
&\quad \mathbf{U}^{\rm E}_*:=\left(\rho^{\rm E}_*,\ \rho^{\rm E}_*u_*,\ \rho^{\rm E}_*v_{J+1,K},\ \frac{p^{\rm E}_*}{\gamma-1}+\frac{\rho^{\rm E}_*}{2}\left[(u_*)^2+(v_{J+1,K})^2\right]\right)^T,\\
&\quad \mathbf{U}^{\rm E}_{J,K}=\begin{cases}\mathbf{U}^{\rm E}_*, & \text{if } u_*+c^{\rm E}_*>0,\\ \overline{\mathbf{U}}_{J+1,K}, & \text{otherwise,}\end{cases}\\
&\texttt{else}\\
&\quad \mathbf{U}^{\rm E}_*:=\left(\rho^{\rm E}_*,\ \rho^{\rm E}_*u_*,\ \rho^{\rm E}_*v_{\rm gh},\ \frac{p^{\rm E}_*}{\gamma-1}+\frac{\rho^{\rm E}_*}{2}\left[(u_*)^2+(v_{\rm gh})^2\right]\right)^T,\\
&\quad \overline{\mathbf{U}}_{\rm gh}=\left(\rho_{\rm gh},\ \rho_{\rm gh}u_{\rm gh},\ \rho_{\rm gh}v_{\rm gh},\ \frac{p_{\rm gh}}{\gamma-1}+\frac{\rho_{\rm gh}}{2}\left[(u_{\rm gh})^2+(v_{\rm gh})^2\right]\right)^T,\\
&\quad \mathbf{U}^{\rm E}_{J,K}=\begin{cases}\mathbf{U}^{\rm E}_*, & \text{if } u_*-c^{\rm E}_*<0,\\ \overline{\mathbf{U}}_{\rm gh}, & \text{otherwise.}\end{cases}
\end{aligned}
$$

The y-direction (situations corresponding to Figures 3.1 (c) and (d)) is treated in a similar fashion, so that $\mathbf{U}^{\rm S}_{J,K}$ are obtained by:

$$
\begin{aligned}
&\texttt{if } v_*>0 \texttt{ then}\\
&\quad \mathbf{U}^{\rm S}_*:=\left(\rho^{\rm S}_*,\ \rho^{\rm S}_*u_{J,K-1},\ \rho^{\rm S}_*v_*,\ \frac{p^{\rm S}_*}{\gamma-1}+\frac{\rho^{\rm S}_*}{2}\left[(u_{J,K-1})^2+(v_*)^2\right]\right)^T,\\
&\quad \mathbf{U}^{\rm S}_{J,K}=\begin{cases}\mathbf{U}^{\rm S}_*, & \text{if } v_*-c^{\rm S}_*<0,\\ \overline{\mathbf{U}}_{J,K-1}, & \text{otherwise,}\end{cases}\\
&\texttt{else}\\
&\quad \mathbf{U}^{\rm S}_*:=\left(\rho^{\rm S}_*,\ \rho^{\rm S}_*u_{\rm gh},\ \rho^{\rm S}_*v_*,\ \frac{p^{\rm S}_*}{\gamma-1}+\frac{\rho^{\rm S}_*}{2}\left[(u_{\rm gh})^2+(v_*)^2\right]\right)^T,\\
&\quad \overline{\mathbf{U}}_{\rm gh}=\left(\rho_{\rm gh},\ \rho_{\rm gh}u_{\rm gh},\ \rho_{\rm gh}v_{\rm gh},\ \frac{p_{\rm gh}}{\gamma-1}+\frac{\rho_{\rm gh}}{2}\left[(u_{\rm gh})^2+(v_{\rm gh})^2\right]\right)^T,\\
&\quad \mathbf{U}^{\rm S}_{J,K}=\begin{cases}\mathbf{U}^{\rm S}_*, & \text{if } v_*+c^{\rm S}_*>0,\\ \overline{\mathbf{U}}_{\rm gh}, & \text{otherwise,}\end{cases}
\end{aligned}
$$

where $\rho_{\rm gh}=\overline{\rho}_{J,K-1},\ u_{\rm gh}=u_{\rm B}(t),\ v_{\rm gh}=2v_{\rm B}(t)-v_{J,K-1},\ p_{\rm gh}=p_{J,K-1}$. Finally, $\mathbf{U}^{\rm N}_{J,K}$

are computed as follows:

`if` $v_*<0$ `then`

$$\mathbf{U}_*^{\mathrm{N}} := \left(\rho_*^{\mathrm{N}},\ \rho_*^{\mathrm{N}}u_{J,K+1},\ \rho_*^{\mathrm{N}}v_*,\ \frac{p_*^{\mathrm{N}}}{\gamma-1}+\frac{\rho_*^{\mathrm{N}}}{2}\left[(u_{J,K+1})^2+(v_*)^2\right]\right)^T,$$

$$\mathbf{U}_{J,K}^{\mathrm{N}} = \begin{cases} \mathbf{U}_*^{\mathrm{N}}, & \text{if } \ v_*+c_*^{\mathrm{N}}>0, \\ \overline{\mathbf{U}}_{J,K+1}, & \text{otherwise,} \end{cases}$$

`else`

$$\mathbf{U}_*^{\mathrm{N}} := \left(\rho_*^{\mathrm{N}},\ \rho_*^{\mathrm{N}}u_{\mathrm{gh}},\ \rho_*^{\mathrm{N}}v_*,\ \frac{p_*^{\mathrm{N}}}{\gamma-1}+\frac{\rho_*^{\mathrm{N}}}{2}\left[(u_{\mathrm{gh}})^2+(v_*)^2\right]\right)^T,$$

$$\overline{\mathbf{U}}_{\mathrm{gh}} = \left(\rho_{\mathrm{gh}},\ \rho_{\mathrm{gh}}u_{\mathrm{gh}},\ \rho_{\mathrm{gh}}v_{\mathrm{gh}},\ \frac{p_{\mathrm{gh}}}{\gamma-1}+\frac{\rho_{\mathrm{gh}}}{2}\left[(u_{\mathrm{gh}})^2+(v_{\mathrm{gh}})^2\right]\right)^T,$$

$$\mathbf{U}_{J,K}^{\mathrm{N}} = \begin{cases} \mathbf{U}_*^{\mathrm{N}}, & \text{if } \ v_*-c_*^{\mathrm{N}}<0, \\ \overline{\mathbf{U}}_{\mathrm{gh}}, & \text{otherwise,} \end{cases}$$

where $\rho_{\mathrm{gh}}=\overline{\rho}_{J,K+1},\ u_{\mathrm{gh}}=u_{\mathrm{B}}(t),\ v_{\mathrm{gh}}=2v_{\mathrm{B}}(t)-v_{J,K+1},\ p_{\mathrm{gh}}=p_{J,K+1}$.

Equipped with all the point values required in (3.1), the cell averages $\{\overline{\mathbf{U}}_{j,k}\}$ in the interior cells are evolved from time t to time $t+\Delta t$. By that time, the boundary will move and some boundary cells may become internal. Assume that the status of the boundary cell (J,K) has changed from boundary to internal. We then need to have approximate solution values there, that is, we need to set the cell averages $\overline{\mathbf{U}}_{J,K}(t+\Delta t)$. This is done according to the following algorithm: the cell averages $\overline{\mathbf{U}}_{J,K}(t+\Delta t)$ are obtained as a weighted average of the intermediate states of the solutions of the aforementioned 1-D Riemann problems — $\mathbf{U}_*^{\mathrm{W}}$, $\mathbf{U}_*^{\mathrm{E}}$, $\mathbf{U}_*^{\mathrm{S}}$, and $\mathbf{U}_*^{\mathrm{N}}$:

$$\overline{\mathbf{U}}_{J,K}(t+\Delta t) := \frac{\alpha^x}{\alpha^x+\alpha^y}\cdot\frac{\alpha^{\mathrm{E}}\mathbf{U}_*^{\mathrm{E}}+\alpha^{\mathrm{W}}\mathbf{U}_*^{\mathrm{W}}}{\alpha^{\mathrm{E}}+\alpha^{\mathrm{W}}}+\frac{\alpha^y}{\alpha^x+\alpha^y}\cdot\frac{\alpha^{\mathrm{N}}\mathbf{U}_*^{\mathrm{N}}+\alpha^{\mathrm{S}}\mathbf{U}_*^{\mathrm{S}}}{\alpha^{\mathrm{N}}+\alpha^{\mathrm{S}}},$$

where

$$\alpha^{\mathrm{E(W)}} = \begin{cases} 1, & \text{if cell } (J\pm1,K) \text{ was internal at time } t, \\ 0, & \text{otherwise,} \end{cases}$$

$$\alpha^{\mathrm{N(S)}} = \begin{cases} 1, & \text{if cell } (J,K\pm1) \text{ was internal at time } t, \\ 0, & \text{otherwise,} \end{cases}$$

and

$$\alpha^x=\max(\alpha^{\mathrm{E}},\alpha^{\mathrm{W}}), \qquad \alpha^y=\max(\alpha^{\mathrm{N}},\alpha^{\mathrm{S}}).$$

Notice that this approach would require every boundary cell to have at least one neighboring internal cell at every time moment.

This completes a description of one evolution step of the proposed 2-D Eulerian method.

4. Two Important Remarks

The proposed boundary cell treatment procedures use the $\mathcal{O}(\Delta x)$ accurate information on the location of the boundary instead of the exact one and thus the *overall accuracy of the introduced method cannot be higher than the first order* (the numerical convergence rate is measured in Example 2 below). We would like to emphasize that the main advantage of the new Eulerian method is its simplicity and robustness. To achieve a higher resolution, one may use an adaptive mesh refinement technique near the boundary, see, e.g., [2, 3]. The adaptive version of the proposed method will be developed in future work.

We would also like to point out that our boundary treatment procedure is not conservative. However, the conservation error occurs only along the boundary surface of co-dimension one. Thus the *total conservation error is expected to be proportional to the mesh size*, as confirmed by our numerical experiments (see Examples 2–7).

5. Numerical Experiments

We test the proposed method on a number of 1-D and 2-D problems. In all the examples below, we apply the semi-discrete second-order central-upwind schemes [13, 14] to the compressible Euler equations for the ideal gas with $\gamma=1.4$. Away from the boundary, the method employs either the 1-D generalized minmod reconstruction (2.5)–(2.6) or its 2-D extension (3.2)–(3.4). In Examples 1 and 3–7, we take $\theta=1.3$, while in Example 2, θ is taken to be 1 in order to minimize the oscillations and avoid appearance of negative pressure values. In the example with a steady geometry (Example 3), the time integration is performed using the third-order strong stability preserving Runge-Kutta method [10], while in the rest of the examples, in which the boundary moves, we have used the forward Euler method.

The main goal of the presented numerical experiments is to demonstrate the robustness of the simple Eulerian treatment of moving boundaries. It should be noticed that near the boundary, the proposed method is only first-order accurate and, in the 1-D case, it can hardly compete with higher-order methods, see, e.g., [7, 8]. The advantage of our approach, however, is in its simple extension to a multiple number of space dimensions as well as its capability to capture complicated multidimensional waves and resolve their interactions.

5.1. One-Dimensional Examples. We begin with two 1-D tests, in which we numerically study a gas in a tube with a moving right boundary located at $x=x_{\rm B}(t)$.

Example 1 — Piston Problem. We first consider a piston problem, taken from [8] (see also [7]). The tube is bounded by a stationary solid wall at $x=0$ and a piston at $x=x_{\rm B}(t)$, which is free to move without friction in the tube. The motion of the piston is governed by the Newton equation:

$$\frac{d^2x_{\rm B}(t)}{dt^2}=\frac{A}{m}\,(p(x_{\rm B}(t),t)-p_{\rm out}),$$

where A is the area of the piston, m is its mass, and $p_{\rm out}$ is an external pressure.

We obtain the velocity of the piston needed for the boundary treatment procedure by numerically solving the following ODE:

$$\frac{d\dot{x}_{\rm B}(t)}{dt}=\frac{A}{m}\,(p_*-p_{\rm out}),$$

where p_* is given by either (2.8) or (2.9), depending on the relation between the current velocity of the piston, $\dot{x}_{\rm B}(t)$, and u_{J-1}, see §2.1.

We take $A/m=2$ and $p_{\text{out}}=2$ and assume that both the gas in the tube and the piston are initially at rest ($u(x,0)\equiv 0$, $\dot{x}_{\text{B}}(0)=0$) and its density and pressure are constant ($\rho(x,0)\equiv p(x,0)\equiv 1$). The piston initial position is $x_{\text{B}}(0)=1$.

Due to the difference between the interior and exterior pressure, the piston starts moving into the tube, rising the pressure inside it. When the pressure in the tube becomes larger than the outside one, the piston decelerates and turns back. The pattern repeats and the motion of the piston becomes oscillatory. In Figure 5.1, we plot the piston trajectory as a function of time computed on four different uniform grids with $\Delta x=1/100$, $1/200$, $1/400$, and $1/2000$ (the computational domain is $[0,1]$). These results demonstrate the ability of our method to accurately capture the piston movement. The obtained numerical solution agrees well with the solution obtained in [8]. We note that the piston problem is different from other examples considered in the paper since the piston motion is not a-priori prescribed but depends on the difference between the pressures inside and outside the tube. However, when the boundary cell extrapolation procedure is used, the elastic properties of the piston are neglected and the wall (piston) is assumed to be solid.

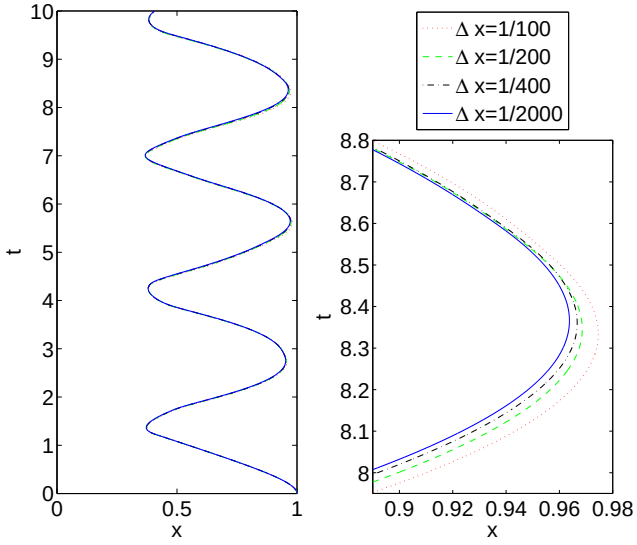

FIG. 5.1. The piston trajectory as a function of time (left) computed on four different uniform grids with $\Delta x=1/100$, $1/200$, $1/400$, and $1/2000$. Zoom at $[0.89,0.98]\times[7.95,8.8]$ (right).

Example 2 — Solid Moving Boundary. In the second 1-D example, we consider a different situation, which better fits our Eulerian approach: we assume that the right boundary of the tube is a solid wall oscillating about $x_{\text{B}}(0)=0.9$ according to the a-priori prescribed equation of motion

$$x_{\text{B}}(t)=0.9+0.1\sin(10\pi t),$$

which, unlike in the case of a piston, is independent of the fluid data. The tube is assumed to be infinitely long on the left (the computational domain is set to $[0,1]$, but the final times are taken sufficiently small so that no waves reach the left end of

the domain). The initial data correspond to a right moving shock, initially positioned at $x=0.5$:

$$(\rho,u,p)=\begin{cases}(4/3,35/99,1.5), & x<0.5,\\ (1,0,1), & 0.5<x<x_{\rm B}(0).\end{cases} \tag{5.1}$$

In Figures 5.2 and 5.3, we show the solution (density and velocity) computed on reasonably coarse uniform spatial grids with $\Delta x=1/100$ and $1/200$ and on a very fine mesh with $\Delta x=1/51200$ at different times. The shock and rarefaction waves, generated by the moving boundary, interact with the incoming shock. This produces a quite complicated solution, whose global structure is accurately resolved by the proposed simple Eulerian method.

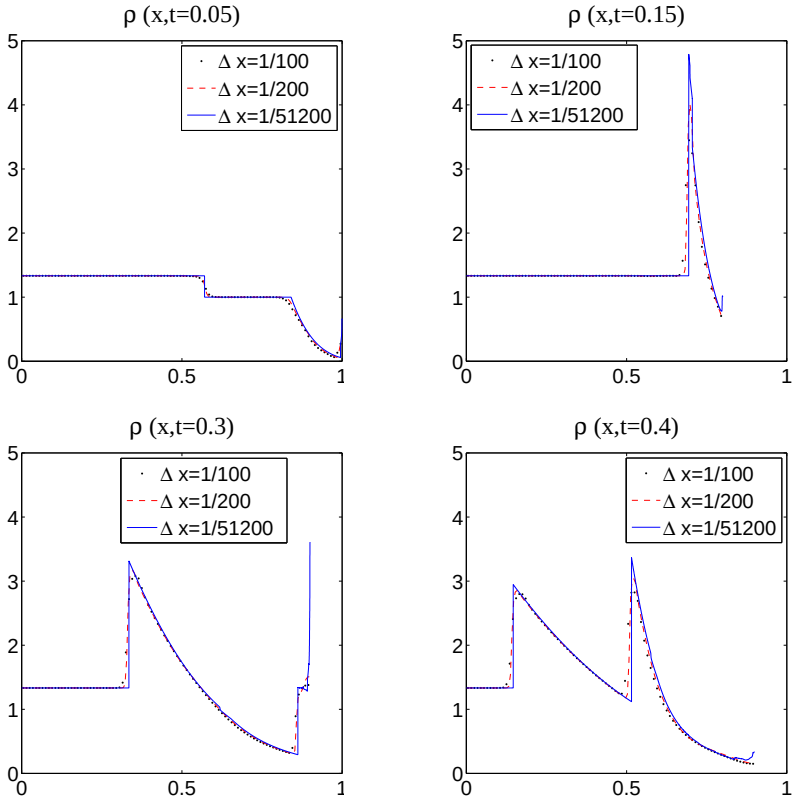

FIG. 5.2. Example 2: Solution (density) computed at times $t=0.05,0.15,0.3$, and 0.4.

As one may observe in Figure 5.2, the local error seems to be larger near the moving boundary than inside the computational domain. To study numerical convergence of our method we check the self-convergence of the computed solutions. We use the high resolution solution (with $\Delta x=1/51200$) as a reference solution and measure the L^1- and local L^1-errors for a sequence of numerical approximations of ρ with $\Delta x=1/100$, $1/200$, $1/400$, $1/800$, and $1/1600$. The obtained errors and numerical orders of convergence are shown in Tables 5.1–5.4 (the local L^1-errors were computed

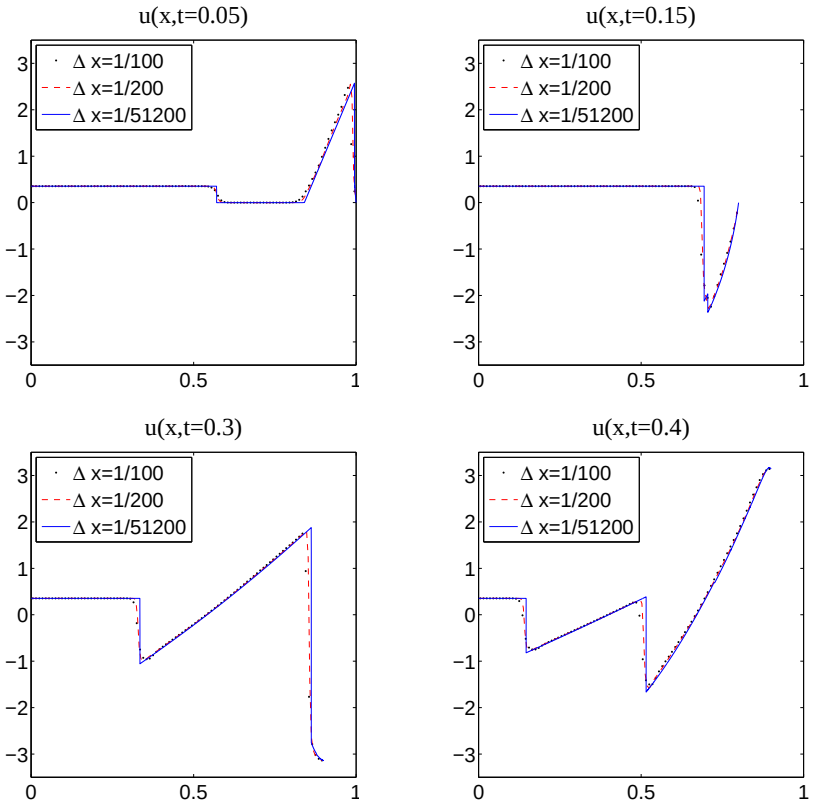

FIG. 5.3. Example 2: Solution (velocity) computed at times $t=0.05, 0.15, 0.3$, and 0.4.

on the intervals $[x_B(t)-0.1, x_B(t)]$). We note that the global L^1 convergence rate of the proposed method is in the range of 0.79–1.05 (Tables 5.2 and 5.4). At times 0.05, 0.15, and 0.3, the local convergence rates (Tables 5.1 and 5.3) are about the same as the global ones, while at time $t=0.4$ the local convergence rate deteriorates to about 0.49 (see Table 5.3). We believe that this occurs because the boundary layer appearing near $x=0.9$ at this time is not resolved by low resolution computations. Since the exact solution of the initial-boundary value problem (2.1), (5.1) is unavailable, we take a closer look at what happens at the right part of the domain. In Figure 5.4, we plot the density $\rho(x,t=0.4)$ computed on 7 uniform grids with $\Delta x = 1/100$, $1/200$, $1/400$, $1/800$, $1/1600$, 3200, and $1/51200$. To improve the resolution achieved near the moving boundary, the mesh might be adaptively refined in those areas (an adaptive implementation of the proposed simple Eulerian method will be explored in future work).

Finally, we measure the mass conservation errors and report them in Table 5.5. As expected, these errors are relatively small and decrease as the mesh is refined, exhibiting the numerical convergence roughly of order one.

5.2. Two-Dimensional Examples. We now turn to 2-D test problems, in which we simulate flow around an (oscillating) solid circle in 2-D. We numerically solve the system (1.1) in the computational domain $[0,1]\times[0,1]$. The fluid domain is

Δx	$t=0.05$		$t=0.15$	
	Error	Rate	Error	Rate
1/100	$5.082\cdot10^{-3}$	–	$2.480\cdot10^{-2}$	–
1/200	$2.666\cdot10^{-3}$	0.93	$1.517\cdot10^{-2}$	0.71
1/400	$1.346\cdot10^{-3}$	0.99	$8.024\cdot10^{-3}$	0.92
1/800	$6.052\cdot10^{-4}$	1.15	$4.376\cdot10^{-3}$	0.87
1/1600	$2.798\cdot10^{-4}$	1.11	$2.434\cdot10^{-3}$	0.85

TABLE 5.1. Example 2: Local L^1-error computed at times $t=0.05$ and $t=0.15$.

Δx	$t=0.05$		$t=0.15$	
	Error	Rate	Error	Rate
1/100	$1.465\cdot10^{-2}$	–	$5.155\cdot10^{-2}$	–
1/200	$7.582\cdot10^{-3}$	0.95	$3.102\cdot10^{-2}$	0.73
1/400	$3.852\cdot10^{-3}$	0.98	$1.710\cdot10^{-2}$	0.86
1/800	$1.856\cdot10^{-3}$	1.05	$9.273\cdot10^{-3}$	0.88
1/1600	$8.959\cdot10^{-4}$	1.05	$4.926\cdot10^{-3}$	0.91

TABLE 5.2. Example 2: L^1-error computed at times $t=0.05$ and $t=0.15$.

given by $[0,1]\times[0,1]\backslash\mathcal{B}_{\rm R}(t)$, where $\mathcal{B}_{\rm R}(t)$ is the rigid ball of radius $R=0.1$ and center $(x_c(t),y_c(t))$, see Figure 5.5. We implement inflow boundary conditions at the left boundary and outflow boundary conditions at the right boundary, while the top and the bottom boundaries of the domain as well as the boundaries of the moving circle are assumed to be solid walls.

We consider a simple rigid movement of the ball with respect to the following equations (a similar problem was considered in [21]):

$$\begin{aligned} &x_c(0)=0.5, \qquad \dot{x}_c(0)=A\omega\cos(\omega t),\\ &y_c(0)=0.5, \qquad \dot{y}_c(0)=B\omega\cos(\omega t), \end{aligned} \tag{5.2}$$

where the constants A and B determine the amplitude of the motion (they are selected sufficiently small so that the ball stays within the computational domain for all times) and ω is its frequency (we take $\omega=10\pi$ in all the examples below).

Example 3 — Moving Shock - Steady Ball. In the first 2-D example, we consider a flow generated by a right moving vertical shock, initially positioned at $x=0.25$ (see Figure 5.5):

$$(\rho(x,y,0),u(x,y,0),v(x,y,0),p(x,y,0))=\begin{cases}(4/3,35/99,0,1.5), & x<0.25,\\ (1,0,0,1), & x>0.25,\end{cases} \tag{5.3}$$

Δx	$t=0.3$		$t=0.4$	
	Error	Rate	Error	Rate
1/100	$1.159\cdot10^{-2}$	–	$3.700\cdot10^{-3}$	–
1/200	$8.239\cdot10^{-3}$	0.49	$3.339\cdot10^{-3}$	0.15
1/400	$4.934\cdot10^{-3}$	0.74	$3.007\cdot10^{-3}$	0.15
1/800	$3.033\cdot10^{-3}$	0.70	$2.168\cdot10^{-3}$	0.47
1/1600	$1.512\cdot10^{-3}$	1.00	$1.542\cdot10^{-4}$	0.49

TABLE 5.3. Example 2: Local L^1-error computed at times $t=0.3$ and $t=0.4$.

Δx	$t=0.3$		$t=0.4$	
	Error	Rate	Error	Rate
1/100	$4.204\cdot10^{-2}$	–	$7.309\cdot10^{-2}$	–
1/200	$2.738\cdot10^{-2}$	0.62	$4.261\cdot10^{-2}$	0.78
1/400	$1.599\cdot10^{-2}$	0.77	$2.465\cdot10^{-2}$	0.79
1/800	$9.057\cdot10^{-3}$	0.82	$1.316\cdot10^{-2}$	0.90
1/1600	$4.727\cdot10^{-3}$	0.94	$7.627\cdot10^{-3}$	0.79

TABLE 5.4. Example 2: L^1-error computed at times $t=0.3$ and $t=0.4$.

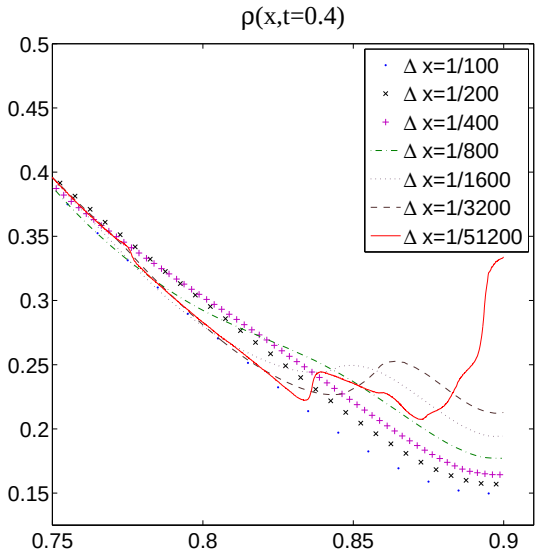

FIG. 5.4. Example 2: Zoom at the solution (density) computed at time $t=0.4$ on different grids.

Δx	$t=0.05$		$t=0.4$	
	Error	Rate	Error	Rate
1/200	$4.371\cdot10^{-3}$	–	$2.580\cdot10^{-3}$	–
1/400	$1.995\cdot10^{-3}$	1.13	$1.548\cdot10^{-3}$	0.74
1/800	$9.673\cdot10^{-4}$	1.04	$8.589\cdot10^{-4}$	0.85
1/1600	$4.565\cdot10^{-4}$	1.08	$4.279\cdot10^{-4}$	1.00
1/3200	$2.202\cdot10^{-4}$	1.05	$2.221\cdot10^{-4}$	0.95

TABLE 5.5. Example 2: Mass conservation errors at times $t=0.05$ and $t=0.4$.

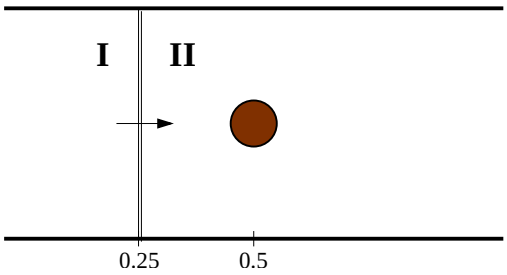

FIG. 5.5. Initial setting for the 2-D numerical examples.

diffracting around the steady ball, that is, $A=B=0$ in (5.2).

The densities, computed at $t=0.2$ and 0.4 using two different uniform meshes of sizes $\Delta x=\Delta y=1/200$ and $\Delta x=\Delta y=1/400$, are plotted in Figure 5.6. One can observe a high resolution achieved by our method despite of its low order boundary treatment.

The mass conservation errors are shown in Table 5.6. As in the 1-D case, these errors decrease as the mesh is refined though the experimental order of convergence is less than one.

$\Delta x=\Delta y$	$t=0.2$		$t=0.4$	
	Error	Rate	Error	Rate
1/100	$1.566\cdot10^{-3}$	–	$1.276\cdot10^{-3}$	–
1/200	$1.070\cdot10^{-3}$	0.55	$1.030\cdot10^{-3}$	0.32
1/400	$5.070\cdot10^{-4}$	1.08	$4.892\cdot10^{-4}$	1.06
1/800	$3.163\cdot10^{-4}$	0.69	$2.900\cdot10^{-4}$	0.75

TABLE 5.6. Example 3: Mass conservation errors at times $t=0.2$ and $t=0.4$.

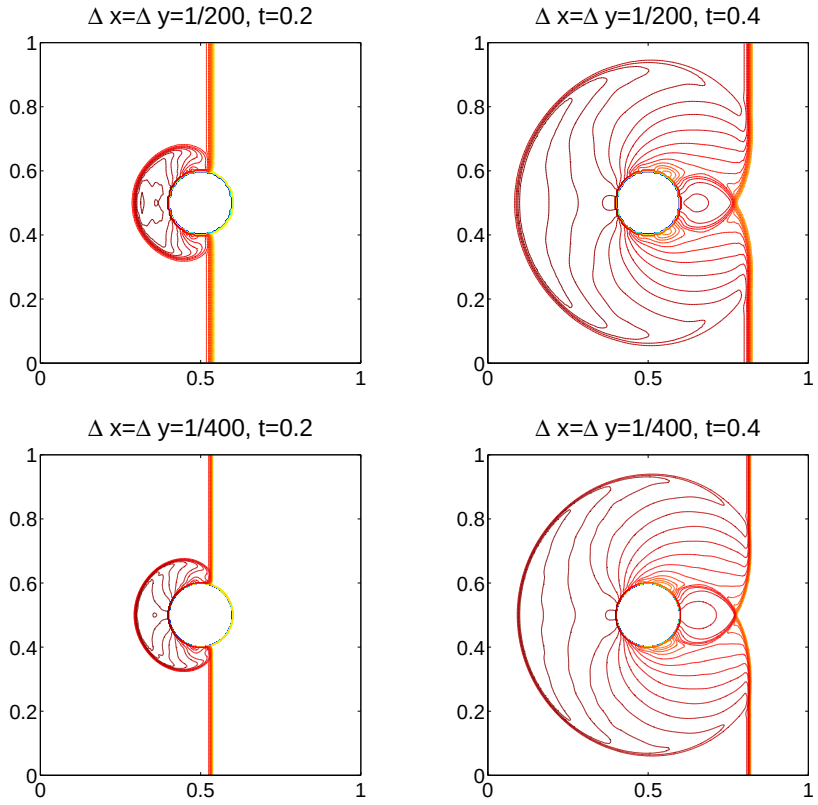

FIG. 5.6. Example 3: Solution (density) computed at times $t=0.2$ and 0.4 on different grids.

Example 4 — No Shock - Slow-Moving Ball. In this example, the ball that moves periodically up and down with a small amplitude ($A=0,\ B=0.01$), is placed in an initially steady flow with

$$\rho(x,y,0)\equiv p(x,y,0)\equiv 1,\quad u(x,y,0)\equiv v(x,y,0)\equiv 0.$$

The ball oscillations are slow so the solution of this problem remains subsonic. Subsequently, the (exact) entropy, defined by p/ρ^γ, is equal to 1 throughout the computational domain and does not change in time. The main goal of this numerical experiment is to check the corresponding behavior of the numerical entropy of the solution, calculated using our method. The L^2-errors in entropy are presented in Table 5.7 together with the mass conservation errors. Even though both quantities decrease relatively slow as the mesh is refined, the results confirm robustness of the proposed simple Eulerian method. In Figure 5.7, we also show the density of the solution computed at time $t=0.1$ using two uniform grids with $\Delta x=\Delta y=1/200$ and $\Delta x=\Delta y=1/400$.

Example 5 — No Shock - Vertically Moving Ball. We now consider the same setting as in Example 4 but with 10 times larger and faster ball oscillations (we take $A=0,\ B=0.1$ in (5.2)). These oscillations generate both shock and rarefection waves of spherical shapes that propagate, interact, and reflects from the top edge of

$\Delta x=\Delta y$	Entropy		Mass Conservation	
	Error	Rate	Error	Rate
1/100	$2.340\cdot10^{-6}$	–	$1.659\cdot10^{-3}$	–
1/200	$1.424\cdot10^{-6}$	0.71	$9.968\cdot10^{-4}$	0.73
1/400	$9.317\cdot10^{-7}$	0.61	$4.853\cdot10^{-4}$	1.04
1/800	$5.663\cdot10^{-7}$	0.72	$2.948\cdot10^{-4}$	0.72

TABLE 5.7. Example 4: L^2 errors in entropy and mass conservation errors at time $t=0.1$.

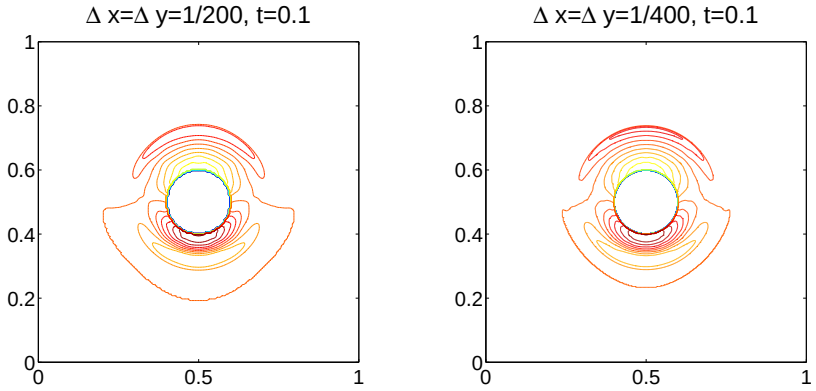

FIG. 5.7. Example 4: Solution (density) computed at time $t=0.1$ on different grids.

the domain. As one can see in Figure 5.8, our method (tested on three uniform grids with $\Delta x=\Delta y=1/200$, $1/400$, and $1/800$) accurately captures the developing solution structure.

The mass conservation errors, reported in Table 5.8, are about 7 times larger than those in the case of the slow-moving bowl, but still decrease with the same order as the mesh is refined.

$\Delta x=\Delta y$	$t=0.1$		$t=0.2$	
	Error	Rate	Error	Rate
1/100	$3.949\cdot10^{-3}$	–	$6.647\cdot10^{-3}$	–
1/200	$2.506\cdot10^{-3}$	0.66	$4.003\cdot10^{-3}$	0.73
1/400	$1.878\cdot10^{-3}$	0.42	$2.778\cdot10^{-3}$	0.53
1/800	$1.017\cdot10^{-3}$	0.89	$1.724\cdot10^{-3}$	0.69

TABLE 5.8. Example 5: Mass conservation errors at times $t=0.1$ and $t=0.2$.

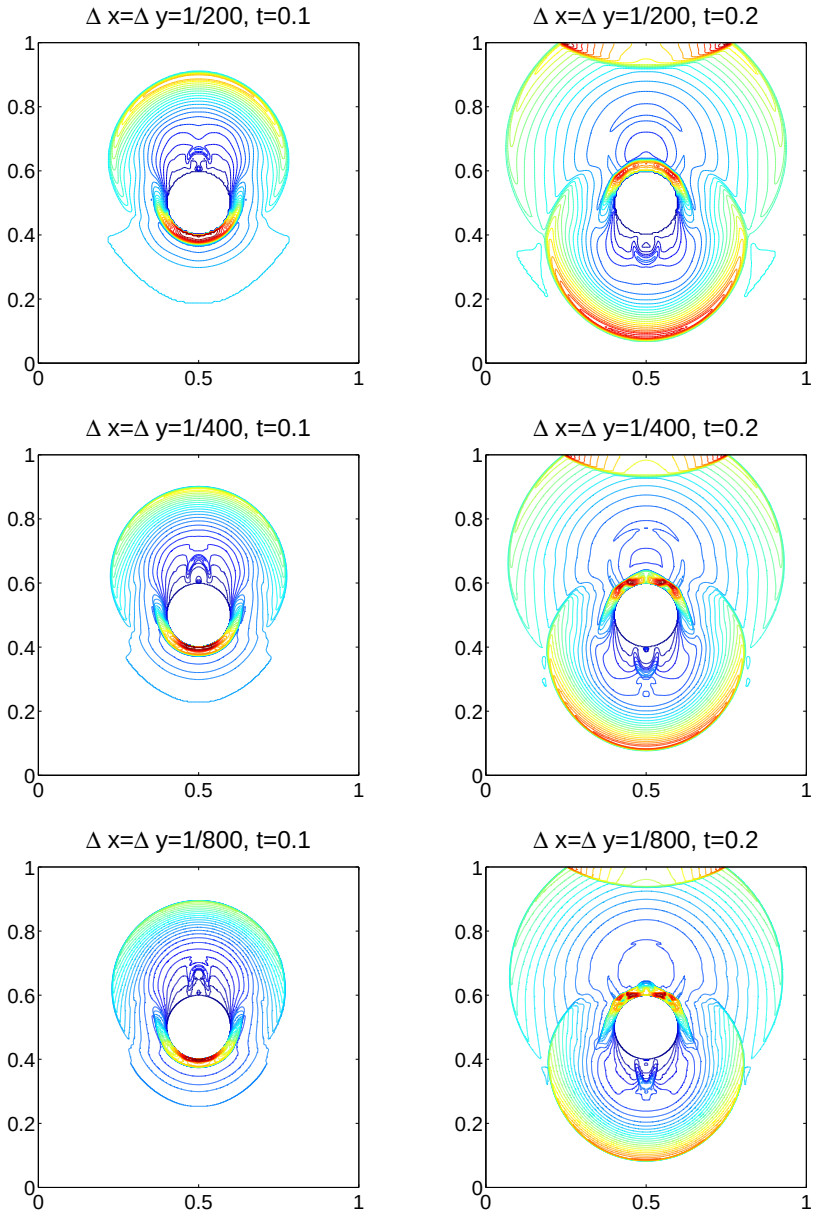

FIG. 5.8. Example 5: Solution (density) computed at times $t=0.1$ and 0.2 on different grids.

Example 6 — No Shock - Diagonally Moving Ball. In order to check the grid dependence of our dimension-by-dimension 2-D method, we modify the previous example by changing the direction of the ball movement from the vertical to a diagonal one (we take $A=B=0.1/\sqrt{2}$ in (5.2)). For small times (until the waves reach the the boundary), the exact solution of this problem is just a rotation of the solution of the problem from Example 5. However, numerical solutions, especially those obtained

using a dimension-by-dimension approach, may be significantly different due to the Cartesian grid effects.

The solutions at time $t=0.1$ computed by the proposed simple Eulerian finite-volume method on three different uniform grids (with $\Delta x=\Delta y=1/200$, $1/400$, and $1/800$) are plotted in Figure 5.9. As one can see, the obtained results are close to the corresponding results presented in the left column of Figure 5.8. The only visible difference is in the low density rarefection region located behind the ball (notice that, at time $t=0.1$, the ball moves in the South-West direction in this example corresponding to the downward direction in Example 5). The conservation errors, reported in Table 5.9, are even smaller and decrease faster than in Example 5.

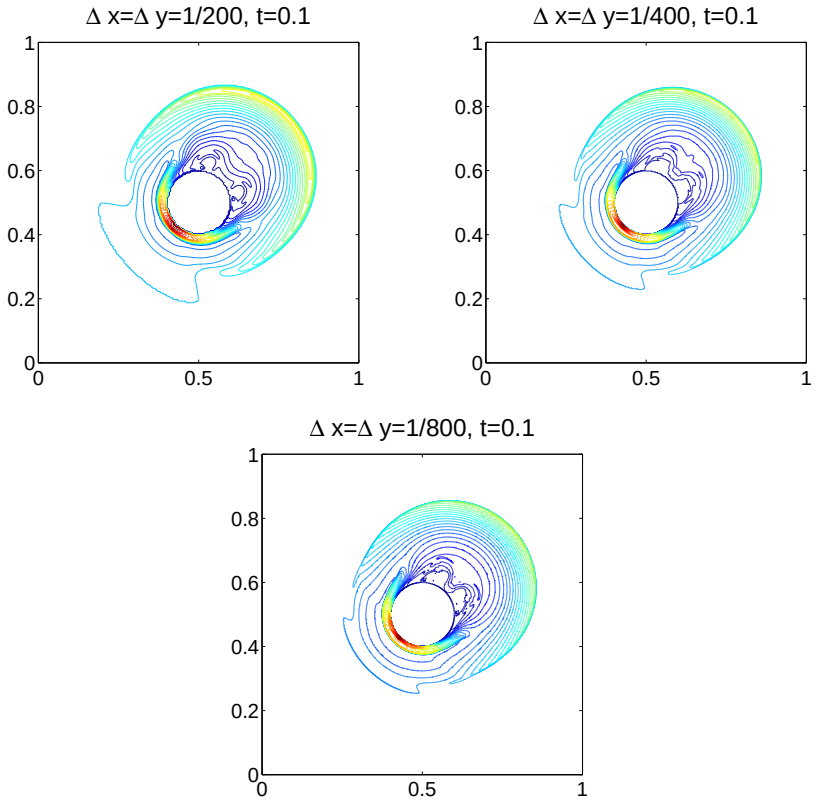

FIG. 5.9. Example 6: Solution (density) computed at time $t=0.1$ on different grids.

Example 7 — Moving Shock - Moving Ball. Finally, we combine the data from Examples 3 and 5 and consider the situation when a shock hits the vertically oscillating ball. We take the coefficients $A=0$, $B=0.1$ in (5.2) and solve the system (1.1) subject to the initial condition (5.3). In this example, the resulting solution structure is even more complicated than in the previous ones. We perform a detailed numerical study of this problem by implementing our method on uniform meshes with $\Delta x=\Delta y=1/200$, $1/400$, and $1/800$. The densities, computed at four different times $t=0.05$, $t=0.1$, $t=0.15$, and 0.2 that correspond to different stages of the ball oscillations, are plotted in Figures 5.10–5.12. The mass conservation errors are

$\Delta x=\Delta y$	$t=0.1$	
	Error	Rate
$1/100$	$5.627\cdot10^{-3}$	–
$1/200$	$3.005\cdot10^{-3}$	0.90
$1/400$	$1.518\cdot10^{-3}$	0.99
$1/800$	$7.032\cdot10^{-4}$	1.11

TABLE 5.9. Example 6: Mass conservation errors at times $t=0.1$.

reported in Tables 5.10–5.11. As in the simpler Examples 3 and 5, the proposed method still achieves a superb resolution.

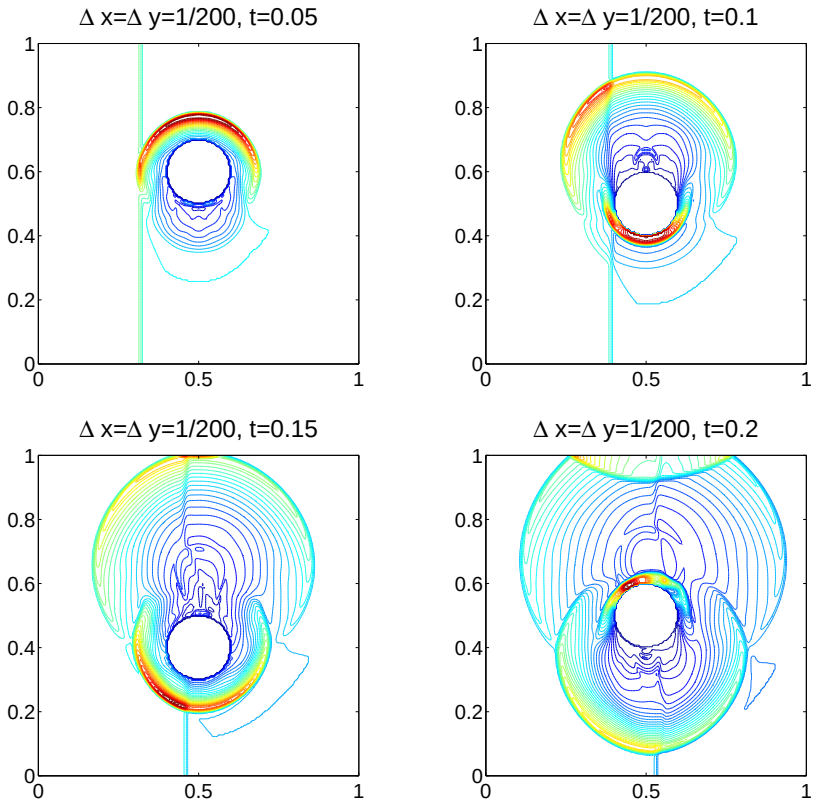

FIG. 5.10. Example 7: Solution (density) computed on a grid with $\Delta x=\Delta y=1/200$.

Acknowledgment. The research of A. Chertock was supported in part by the NSF Grants DMS-0410023 and DMS-0712898. The research of A. Kurganov was supported in part by the NSF Grant DMS-0610430.

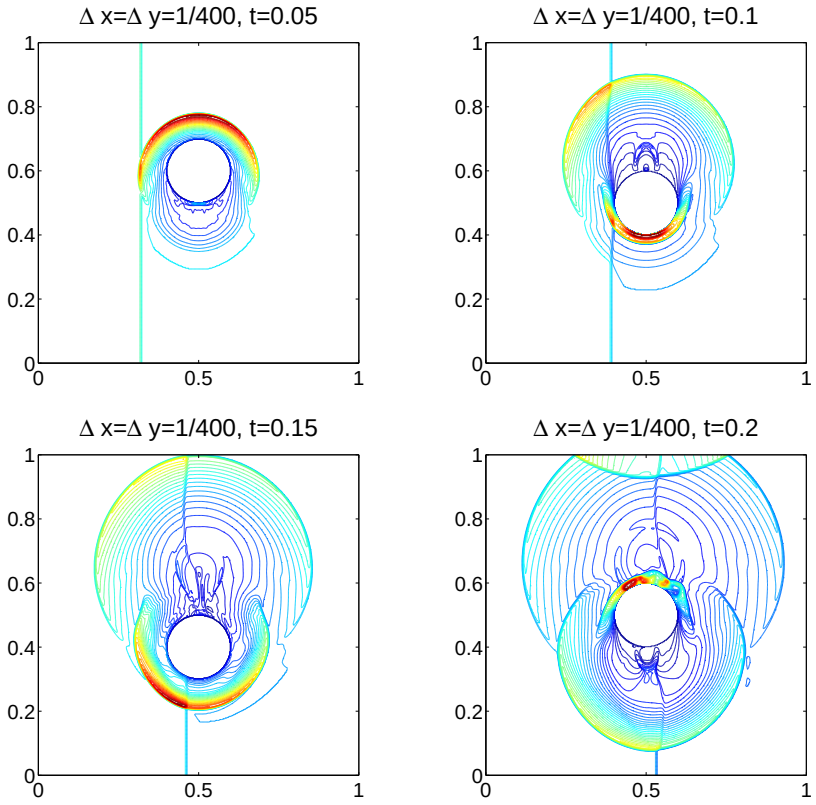

FIG. 5.11. Example 7: Solution (density) computed on a grid with $\Delta x=\Delta y=1/400$.

$\Delta x=\Delta y$	$t=0.05$		$t=0.1$	
	Error	Rate	Error	Rate
$1/100$	$5.056\cdot10^{-4}$	–	$3.984\cdot10^{-3}$	–
$1/200$	$2.453\cdot10^{-4}$	1.04	$2.513\cdot10^{-3}$	0.66
$1/400$	$2.105\cdot10^{-4}$	0.22	$1.879\cdot10^{-3}$	0.42
$1/800$	$9.038\cdot10^{-5}$	1.22	$1.017\cdot10^{-3}$	0.89

TABLE 5.10. Example 7: Mass conservation errors at times $t=0.05$ and $t=0.1$.

REFERENCES

[1] Bassi, F. and Rebay, S., *High-order accurate discontinuous finite element solution of the 2D Euler equations*, J. Comput. Phys., 138, 251–285, 1997.

[2] M.J. Berger and P. Colella, *Local adaptive mesh refinement for shock hydrodynamics*, J. Comput. Phys., 82, 67–84, 1989.

[3] M.J. Berger and J. Oliger, *Adaptive mesh refinement for hyperbolic partial differential equations*, J. Comput. Phys., 53, 482–512, 1984.

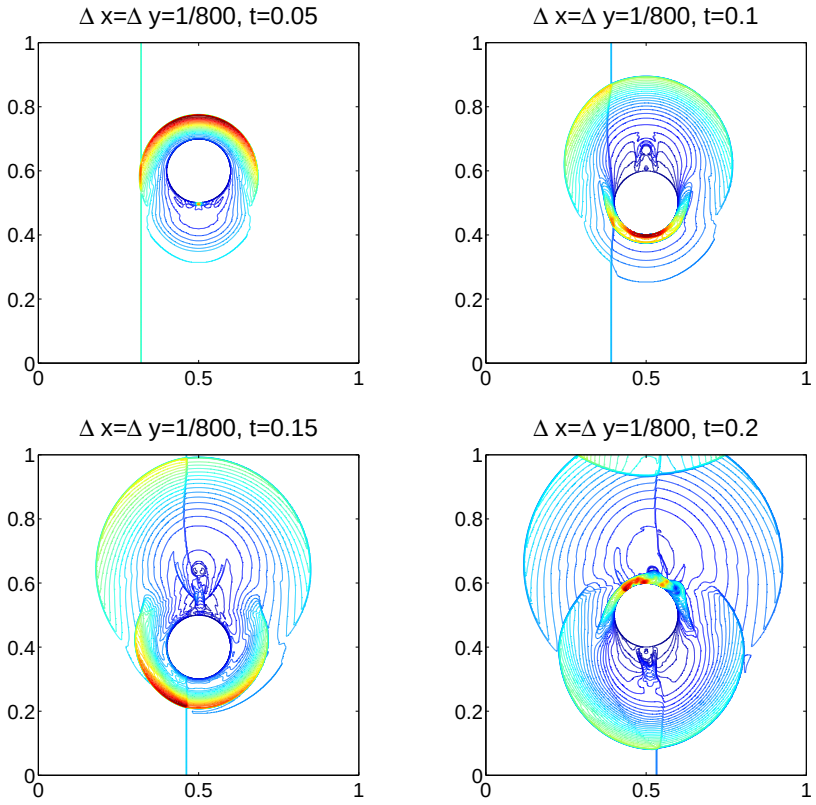

FIG. 5.12. Example 7: Solution (density) computed on a grid with $\Delta x = \Delta y = 1/800$.

$\Delta x = \Delta y$	$t = 0.15$		$t = 0.2$	
	Error	Rate	Error	Rate
$1/100$	$3.505 \cdot 10^{-3}$	–	$7.979 \cdot 10^{-3}$	–
$1/200$	$2.096 \cdot 10^{-3}$	0.74	$4.410 \cdot 10^{-3}$	0.86
$1/400$	$1.462 \cdot 10^{-3}$	0.52	$2.992 \cdot 10^{-3}$	0.56
$1/800$	$8.112 \cdot 10^{-4}$	0.85	$1.857 \cdot 10^{-3}$	0.69

TABLE 5.11. Example 7: Mass conservation errors at times $t = 0.15$ and $t = 0.2$.

[4] Dadone A, *Symmetry techniques for the numerical solution of the 2D Euler equations at impermeable boundaries*, Internat. J. Numer. Methods Fluids, 28, 1093–1108, 1998.
[5] A. Chertock, S. Karni, and A. Kurganov, *Interface tracking method for compressible multifluids*, M2AN Math. Model. Numer. Anal., submitted.
[6] S.F. Davis, *An interface tracking method for hyperbolic systems of conservation laws*, Appl. Numer. Math., 10, 447-472, 1992.
[7] R. Fazio and R.J. LeVeque, *Moving-mesh methods for one-dimensional hyperbolic problems using CLAWPACK*, Comput. Math. Appl. 45, 273–298, 2003.
[8] R. Fazio and G. Russo, *A Lagrangian central scheme for multi-fluid flows. Hyperbolic problems:*

theory, numerics, applications, Vol. I, II (Magdeburg, 2000), 347-356, Internat. Ser. Numer. Math., 140, 141, Birkhuser, Basel, 2001.

[9] E. Godlewski and P.-A. Raviart, *Numerical approximation of hyperbolic systems of conservation laws*, Springer-Verlag, New York, 1996.

[10] S. Gottlieb, C.-W. Shu, and E. Tadmor, *High order time discretization methods with the strong stability property*, SIAM Review, 43, 89-112, 2001.

[11] L. Krivodonova and M. Berger, *High-order accurate implementation of solid wall boundary conditions in curved geometries*, J. Comput. Phys., 211, 492–512, 2006.

[12] D. Kröner, *Numerical Schemes for Conservation Laws*, Wiley, Chichester, 1997.

[13] A. Kurganov and C.-T. Lin, *On the reduction of numerical dissipation in central-upwind schemes*, Commun. Comput. Phys., 2, 141-163, 2007.

[14] A. Kurganov, S. Noelle, and G. Petrova, *Semi-discrete central-upwind schemes for hyperbolic conservation laws and Hamilton-Jacobi equations*, SIAM J. Sci. Comput., 23, 707-740, 2001.

[15] A. Kurganov and E. Tadmor, *New high-resolution central schemes for nonlinear conservation laws and convection-diffusion equations*, J. Comput. Phys., 160, 214-282, 2000.

[16] B. van Leer, *Towards the ultimate conservative difference scheme, V. A second order sequel to Godunov's method*, J. Comput. Phys., 32, 101-136, 1979.

[17] R. LeVeque, *Finite volume methods for hyperbolic problems*, Cambridge Texts in Applied Mathematics, Cambridge University Press, 2002.

[18] K.-A. Lie and S.Noelle, *On the artificial compression method for second-order nonoscillatory central difference schemes for systems of conservation laws*, SIAM J. Sci. Comput., 24, 1157-1174, 2003.

[19] H. Nessyahu and E. Tadmor, *Non-oscillatory central differencing for hyperbolic conservation laws*, J. Comput. Phys., 87, 408-463, 1990.

[20] P. K. Sweby, *High resolution schemes using flux limiters for hyperbolic conservation laws*, SIAM J. Numer. Anal., 21, 995-1011, 1984.

[21] D. Teleaga and J. Struckmeier, *A finite-volume particle method for conservation laws on moving domains*, Internat. J. Numer. Methods Fluids, to appear.

[22] E.F. Toro, *Riemann solvers and numerical methods for fluid dynamics. A practical introduction*, Second edition, Springer-Verlag, Berlin, 1999.