Development of a CFD Solver For Primary Diesel Jet Atomization in FOAM-Extend

Vuko Vukčević, Robert Keser, Hrvoje Jasak

University of Zagreb, Croatia

Michele Battistoni

University of Perugia, Italy

Hong Im

King Abdullah University of Science and Technology, Saudi Arabia

Johan Roenby

Stromning, Denmark

Copyright © 2019 Society of Automotive Engineers, Inc.

ABSTRACT

Ongoing development of a CFD framework for the simulation of primary atomization of a high pressure diesel jet is presented in this work. The numerical model is based on a second order accurate, polyhedral Finite Volume method implemented in `foam-extend-4.1`, a community driven fork of the OpenFOAM software. A geometric VOF method `isoAdvector` is used for interface advection, while the Ghost Fluid Method (GFM) is used to handle the discontinuity of the pressure and the pressure gradient at the interface between the two phases: n-dodecane and air in the combustion chamber. In order to obtain highly resolved interface while minimizing computational time, an Adaptive Grid Refinement (AGR) strategy for arbitrary polyhedral cells is employed in order to refine the parts of the grid near the interface. Dynamic Load Balancing (DLB) is used in order to preserve parallel efficiency during AGR. The combination of `isoAdvector`–GFM–AGR–DLB presents a unique framework for diesel jet atomization. The developed numerical framework is preliminarily tested on the Spray D geometry. The unstructured, mostly hexahedral grid is used with the base cell size of 40 micrometres. Four refinement levels are used in the close proximity of the interface in order to attempt to resolve break-up of droplets. The finest cells near the interface have the size of 2.5 micrometres. Part of the nozzle is also considered in the simulation in order to capture the developed jet profile at the entry into the combustion chamber. The temporal evolution of the jet is presented, along with the preliminary comparison of droplet statistics with available results.

INTRODUCTION

The multi-physics nature of the combustion process happening in a direct Compression Ignition (CI) engine makes

a Direct Numerical Simulation (DNS) of a whole engine cycle close to impossible with present resources. This leads to development of numerical models for the spray utilizing either the Lagrangian–Eulerian framework [1, 2] or Eulerian–Eulerian framework [3]. Both Lagrangian–Eulerian or Eulerian–Eulerian numerical frameworks require sub-models to capture the unresolved physics, whether it is the primary or secondary break-up, coalescence or evaporation.

With present computational resources and high-fidelity CFD codes, the DNS of the primary break-up is slowly beginning to become feasible. The computational resources required to perform such computations are still prohibitively large from the engineering point of view, but may provide rich information on the primary break-up, which can readily be used to develop better sub-models for the engineering type CFD codes. As an early example, Desjardins *et al.* [4] developed a conservative LS/GFM and applied it to turbulent atomization of a liquid diesel jet, where the Reynolds and Weber numbers have been reduced to make the direct simulation possible at the time (2008). The computational grid had approximately 17 million elements, providing quite rich flow details. Another set of high-fidelity numerical simulations has been performed by Ghiji *et al.* [5] in their combined numerical and experimental analysis of early stage diesel sprays. They have shown that the first break-up pattern obtained with their VOF code qualitatively matches the one observed in the experiments. They also found that the finer grids lead to smaller droplet sizes, decrease in the early spray angle and increase in the liquid core length. Recently, Arienti and Sussman [6] performed a numerical analysis of the Engine Combustion Network's (ECN) Spray A injector that includes thermal transient effects. Both n-dodecane and air are considered compressible and the simulations are performed with the assumption of adiabatic and isothermal wall conditions with the lifting needle. The simulations provided useful insight on the break-up length vari-

ation ultimately caused by the difference in the surface tension coefficient depending on the varying temperature of the fuel. The simulations also proved to be quite informative on the structures of separated fuel: approximately two thirds of the structures had a nearly spherical shape, while the remaining one third had a more elongated shape (ligaments), which would have probably undergone a secondary break-up eventually. Another state-of-the-art primary break-up simulation has been presented as a part of the Battistoni *et al.*'s review paper [7]. The simulation concerns ECN's Spray D injector and it required 188 000 core hours of computational time to perform, providing insightful physics happening during the atomization. The DNS code is based on a compressible, multiphase semi-implicit method by Jemison *et al.* [8]. The interface is represented with the Coupled Level Set-Volume-Of-Fluid (CLS-VOF) method, while the code employs AGR in order to keep the grid coarse in regions of minor interest far from the interface. The research suggests that the scientific community is constantly pushing the limits of the present-day numerical codes in order to understand the underlying combustion physics in a better way. Following these trends, we present a numerical framework for primary jet atomization based on the open source library `foam-extend-4.1`.

The paper is organized as follows. The next section presents the mathematical model of two incompressible, Newtonian fluids with a sharp interface between them. Following the mathematical model, the numerical model is presented in detail, which contains four parts: i) Handling of interface discontinuities using the Ghost Fluid Method (GFM), ii) Interface advection using the `isoAdvector` algorithm, iii) Polyhedral Adaptive Grid Refinement (AGR) and iv) Parallel Dynamic Load Balancing (DLB). The numerical modelling section also presents the solution algorithm of the combined framework. The ongoing research regarding the ECN's Spray D injector are presented next. The paper is concluded with a discussion and a pathway to future work.

MATHEMATICAL MODEL

We consider two incompressible, Newtonian fluids separated by a sharp interface in a gravitational field. The motion of each fluid is governed by Navier–Stokes equations in primitive form together with the incompressibility constraint [4]:

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u}\mathbf{u}) - \nabla \cdot (\nu \nabla \mathbf{u}) = -\frac{1}{\rho} \nabla p_d, \quad (1)$$

$$\nabla \cdot \mathbf{u} = 0. \quad (2)$$

Here $\mathbf{u}$ is the velocity field, and ν is the kinematic viscosity field assumed to take different constant values, ν^+ and ν^- , in each of the two fluids. Similarly, ρ is the density field taking different constant values, ρ^+ and ρ^- , in each fluid. Following the conventions from the potential flow and some CFD models, the quantity p_d in (1) is the dynamic pressure defined as the pressure field, p , with the

hydrostatic potential subtracted:

$$p_d = p - \rho \mathbf{g} \cdot \mathbf{x}, \quad (3)$$

where $\mathbf{g}$ is the gravitational acceleration and $\mathbf{x}$ is the position vector. For two-fluid problems we must also account for the position and motion of the fluid interface, on which appropriate boundary conditions must be imposed. In what follows, we will work with a slightly simplified form of jump conditions at the interface, neglecting the effect of tangential stress balance compared to normal stress balance [9]. This assumption is justified for flows with high Reynolds numbers that are of interest in this work, as discussed by Huang *et al.* [10] in detail. The jump conditions are briefly outlined here, while the reader is referred to [9] for a detailed analysis.

- Density discontinuity:

$$[\rho] = \rho^- - \rho^+, \quad (4)$$

where $[\cdot]$ notation is taken from the GFM literature [4, 10] and denotes the jump in variables across the interface. Superscripts $^+$ and $^-$ denote the values infinitesimally close to the interface in heavier and in lighter fluid, respectively.

- Kinematic boundary condition:

$$[\mathbf{u}] = \mathbf{u}^- - \mathbf{u}^+ = \mathbf{0}. \quad (5)$$

Kinematic boundary condition ensures the continuity of the velocity field at the interface.

- Simplified tangential stress balance:

$$[\nabla_n \mathbf{u}_t] = \mathbf{0}, \quad (6)$$

stating that the normal gradient of the tangential velocity field does not have a jump. This simplified form is obtained by neglecting surface divergence of surface tension force and surface gradient of the normal velocity component [9].

- Dynamic boundary condition:

$$[p_d] = -[\rho] \mathbf{g} \cdot \mathbf{x} - \sigma \kappa, \quad (7)$$

where σ is surface tension coefficient and κ is twice the mean curvature of the interface.

- Additional dynamic boundary condition:

$$\left[\frac{\nabla p_d}{\rho} \right] = \mathbf{0}, \quad (8)$$

follows from the inspection of Navier–Stokes equations ((1)) when one assumes the simplified form of the tangential stress balance given by (6).

It is important to clearly state that the jump conditions given by Eqns. (4)–(8) have been derived with the assumption of high Reynolds numbers. High Reynolds number flows investigated in this work allow us to assume

that the tangential stress balance is of minor importance compared to normal stress balance [10]. The kinematic viscosity is therefore defined in terms of volume fraction function [9]:

$$\nu = \alpha\nu^+ + (1 - \alpha)\nu^-, \quad (9)$$

where ν^+ is the kinematic viscosity of heavier fluid and ν^- is the kinematic viscosity of lighter fluid. In the VOF method, the volume fraction α is defined as:

$$\alpha = \frac{V^+}{V}, \quad (10)$$

where V^+ is the volume occupied by water inside a control volume V . The mass conservation equation for one phase (fluid +) reduces to the well-known VOF advection equation:

$$\frac{\partial \alpha}{\partial t} + \nabla \cdot (\mathbf{u}\alpha) = 0, \quad (11)$$

where solenoidal velocity field has been assumed, as given by (2).

NUMERICAL MODEL

Continuity and Navier–Stokes equations ((2) and (1)) are discretised in space using a second-order accurate, collocated FV method for general unstructured/structured grids [11]. An arbitrary polyhedral control volume (CV) has a number of neighbours, each defined with surface area vector $\mathbf{s}_f$ and distance vector $\mathbf{d}_f$ from cell centre P to neighbouring cell centre N , as shown in Figure 1. The governing equations for the flow field ((2) and (1)) and the interface advection equation ((11)) constitute a nonlinear system of coupled partial differential equations. The pressure–velocity–interface coupling is achieved using a combination of SIMPLE [12] and PISO [13] algorithms, where a number of PISO correctors can be used within each SIMPLE (nonlinear, or outer) correction step to ensure faster convergence without relaxation factors, as discussed by Vukčević *et al.* [14]. The continuity equation is used to derive a dynamic pressure equation using the Rhie–Chow interpolation [15] as a filter for spurious pressure oscillations. The reader is referred to Jasak [11] and Ferziger and Perić [16] for the details regarding polyhedral FV discretisation and the solution algorithms. In the following text, we present the interface-corrected dynamic pressure interpolation using the GFM, isoAdvector method for interface advection and AGR strategy for polyhedral cells combined with DLB.

INTERFACE-CORRECTED DYNAMIC PRESSURE INTERPOLATION WITH THE GHOST FLUID METHOD

One of the fundamental steps in the collocated FV method is the interpolation of fields from cell centres to face centres. Linear interpolation (or central differencing), based on Taylor series expansion becomes erroneous in presence of a discontinuity, since the expansion assumes sufficiently smooth spatial variation (C^1 continuity). This can be easily demonstrated in a simplified, one-dimensional case presented in Figure 2, where P and N denote cell

Figure 1: Polyhedral control volume. Control volume P shares a common face with its immediate neighbour N .

centres, f is the face between them and Γ_f represents the interface where the discontinuity in ϕ and $\nabla\phi$ is present. Simple linear interpolation of cell centred values ϕ_P and ϕ_N to face centred value ϕ_f yields an incorrect value since the discontinuity is not taken into account. The idea behind the GFM is to use one-sided extrapolates to define "ghost" values at the other side of the interface by second-order accurate discretisation of interface jump conditions. Since two equations ((7) and (8)) for dynamic pressure discontinuities exist, one can introduce two additional unknowns: p_d^+ and p_d^- , infinitesimally close to the interface from both sides. These values can be solved for and expressed in terms of cell-centred values (*i.e.* $p_d^+ = p_d^+(p_{dN}, p_{dP})$), providing correct gradients. Using the correct gradients, the second-order extrapolation from the interface towards the neighbouring cell centre is carried out. This procedure is presented in Figure 3 for a general discontinuous variable ϕ , defining one-sided extrapolates respecting the jump conditions at the interface. It is important to stress that no assumption has been made so far on the location of the interface.

The GFM interpolation is only required in the presence of discontinuities, while far from the interface, ordinary interpolation is sufficiently accurate. In the following analysis, we assume that the interface location can be readily estimated from the volume fraction field, provided that it remains sharp.

Consider a computational stencil on polygonal two-dimensional grid for clarity, Figure 4. The interface, denoted by blue dashed line is defined with volume fraction contour $\alpha = 0.5$. Cells with $\alpha > 0.5$ are marked as "wet cells", while the cells with $\alpha < 0.5$ are marked as "dry cells". $\mathbf{x}_\Gamma$ represents the location of the interface somewhere along the distance vector $\mathbf{d}_f$ between adjacent cell centres. The exact location of the interface between P

and N can be defined as:

$$\mathbf{x}_\Gamma = \mathbf{x}_P + \lambda \mathbf{d}_f, \quad (12)$$

where the parametrised distance to the interface λ can be readily estimated using the volume fraction [14]:

$$\lambda = \frac{\alpha_P - 0.5}{\alpha_P - \alpha_N}. \quad (13)$$

Note that such procedure of defining the location of the interface does not require reconstruction using the Level Set signed distance field [10]. Formally, the location estimate given by (13) is of the same order of accuracy as the advection step.

Before discretising the jump conditions at the interface, we introduce a substitution for the inverse density following Huang *et al.* [10]:

$$\beta = \frac{1}{\rho}. \quad (14)$$

The dynamic pressure jump conditions, written in terms of

Figure 2: Inaccurate linear interpolation from cells P and N to face f for a field with discontinuity at the interface Γ_f .

Figure 3: Ghost Fluid Method interpolation from cells P and N to face f for a field with discontinuity at the interface Γ_f .

Figure 4: Unstructured interface stencil in two-dimensions [14]. Red face is an interface face.

inverse density β reads:

$$[p_d] = p_d^- - p_d^+ = \left(\frac{1}{\beta^-} - \frac{1}{\beta^+} \right) \mathbf{g} \cdot \mathbf{x}_\Gamma - \sigma \kappa = \mathcal{H}, \quad (15)$$

After the advection step, the location of the interface is calculated for each pair of the interface cells using (12). The jump in dynamic pressure can then be evaluated explicitly using (15).

The interface curvature κ is calculated with volume fraction field as:

$$\kappa = \frac{\nabla \alpha}{|\nabla \alpha| + \epsilon}, \quad (16)$$

where $\epsilon = 10^{-15}$ is used to stabilise the division in the areas far from the interface. Note that the gradient of volume fraction, $\nabla \alpha$ is evaluated with least squares gradient [17]. Furthermore, the gradient is smoothed by interpolating from cell-centres to cell vertices and back from cell vertices to cell-centres, which proved to decrease the magnitude of parasitic velocities related to curvature calculation. Future work needs to be done in order to quantify this effect and compare it with well-known methods such as the Height Function method presented by Ivey and Moin [18].

Following Huang *et al.* [10] and previous work by Vukčević *et al.* [14], the dynamic pressure gradient jump condition is discretised in a second-order accurate manner using one-sided gradient evaluations based on parametrised distance to the interface (13):

$$[\beta \nabla p_d] = \beta^- (\nabla p_d)^- - \beta^+ (\nabla p_d)^+ = \beta^- \frac{p_{dN} - p_d^-}{1 - \lambda} - \beta^+ \frac{p_d^+ - p_{dP}}{\lambda} = 0. \quad (17)$$

The system of equations given by discretised jump conditions ((15) and (17)) can be easily solved for p_d^+ and p_d^- . Here, the complete procedure for extrapolation is presented for cell P using p_d^+ , while the procedure for cell N using p_d^- is analogous and can be easily inferred.

$$p_d^+ = \frac{\lambda \beta^-}{\bar{\beta}_w} p_{dN} + \frac{(1 - \lambda) \beta^+}{\bar{\beta}_w} p_{dP} - \frac{\lambda \beta^-}{\bar{\beta}_w} \mathcal{H}, \quad (18)$$

where $\bar{\beta}_w$ is the weighted inverse density:

$$\bar{\beta}_w = \lambda \beta^- + (1 - \lambda) \beta^+. \quad (19)$$

It is interesting to note that $\overline{\beta_w}$ actually represents the harmonic interpolation of density based on the actual distance to the interface. Once the p_d^+ is known, the dynamic pressure field is extrapolated from the heavier fluid (fluid “+”) at the location infinitesimally close to the interface, towards the neighbouring cell centre:

$$p_{dN}^+ = p_d^+ + \frac{1-\lambda}{\lambda} (p_d^+ - p_{dP}) . \quad (20)$$

Substituting (18) into (20) yields the extrapolated value at the neighbouring cell centre given in terms of two cell centred values, inverse density and the explicit jump term $\mathcal{H}$:

$$p_{dN}^+ = \frac{\beta^-}{\beta_w} p_{dN} + \left(1 - \frac{\beta^-}{\beta_w}\right) p_{dP} - \frac{\beta^-}{\beta_w} \mathcal{H}, \quad (21)$$

(21) (and analogous expression for p_{dP}^-) are used whenever the discretisation requires cell-centred values from the other side of the interface. The jump conditions at the interface are therefore taken into account for the pressure gradient in the Navier–Stokes equations ((1)) and the pressure Laplacian in the continuity equation ((2)). The procedure has been derived using a compact computational stencil, respecting the collocated FV framework with face-based connectivity [11]. This procedure results in a symmetric discretisation of the Laplacian operator, thus preserving the symmetry of the underlying differential operator. This has been discussed in detail by Vukčević *et al.* [14], while in-depth derivation of the procedure is presented by Vukčević [9], and shall not be repeated here. The proposed method belongs to a family of balanced force methods (see *e.g.* [19]), where the coupling between density field and dynamic pressure is resolved within the pressure equation instead of the momentum equation. The procedure assumes a-priori known location of the interface defined by the volume fraction field α , making it suitable for segregated (or partitioned) solution algorithms as the one used in this work. The method presented so far is at most second-order accurate, although the accuracy will directly depend on the accuracy of the advection step, (11). The next section is devoted to second-order accurate advection of the interface.

INTERFACE ADVECTION WITH THE ISOADVECTOR SCHEME The implicit representation of a fluid interface via volume fractions is the natural one in the FV framework. The task of advancing the interface in time becomes a matter of modelling the composition (heavy and light fluid) of the total volume of fluid passing from one cell into its neighbour during a time step. Typically, the available information consists of volume fractions in cells at the beginning of the time step, α_P , and the velocity field represented in two ways, namely by the cell averaged velocity, $\mathbf{u}_P$, and by the volumetric face flux, ϕ_f . These velocity field representations are available at the beginning of the time step, and since the nonlinear (outer) iterations are performed, we may also have estimates for them at the end of the time step. The challenge of advecting the fluid interface becomes a question of using

$\alpha_P(t)$, $\mathbf{u}_P(t)$ and $\phi_f(t)$, and possibly available estimates of $\mathbf{u}_P(t + \Delta t)$ and $\phi_f(t + \Delta t)$, to predict $\alpha_P(t + \Delta t)$. In the following, we will describe how this task is performed using the *isoAdvector* algorithm by Roenby *et al.* [20]. The starting point of the *isoAdvector* is the continuity equation for the density field integrated over the volume of an interface cell:

$$\frac{d}{dt} \int_V \rho(\mathbf{x}, t) dV + \sum_f \int_{S_f} \rho(\mathbf{x}, t) \mathbf{u}(\mathbf{x}, t) \cdot d\mathbf{S} = 0. \quad (22)$$

Here V is the cell volume, S_f is the surface of one of the faces comprising the cell boundary and the sum $\sum_f$ is over all the cell’s faces. Without loss of generality, we define a normalised and shifted density field, or indicator function as:

$$H(\mathbf{x}, t) = \frac{\rho(\mathbf{x}, t) - \rho^-}{\rho^+ - \rho^-}, \quad (23)$$

where ρ^- and ρ^+ are considered constant. Isolating $\rho(\mathbf{x}, t)$ in (23) and inserting it into (22), after some rearrangement it follows:

$$\begin{aligned} \frac{d}{dt} \int_V H(\mathbf{x}, t) dV + \sum_f \int_{S_f} H(\mathbf{x}, t) \mathbf{u}(\mathbf{x}, t) \cdot d\mathbf{S} \\ = -\frac{\rho^-}{\rho^+ - \rho^-} \sum_f \int_{S_f} \mathbf{u}(\mathbf{x}, t) \cdot d\mathbf{S}. \end{aligned} \quad (24)$$

So far no assumption of incompressibility has been made. Assuming two constants ρ^+ and ρ^- are indeed the densities of the heavy and light fluid, respectively, then both fluids are incompressible, causing the right hand side in (24) to vanish. The indicator function, $H(\mathbf{x}, t)$, becomes a 3-dimensional Heaviside function taking the values 0 and 1 in the region of space occupied by the light and the heavy fluid, respectively. With the definitions of the volume fraction of cell P :

$$\alpha_P = \frac{1}{V_P} \int_{V_P} H(\mathbf{x}, t) dV, \quad (25)$$

(24) can be written as:

$$\frac{d\alpha_P}{dt} + \frac{1}{V_P} \sum_f \int_{S_f} H(\mathbf{x}, t) \mathbf{u}(\mathbf{x}, t) \cdot d\mathbf{S} = 0. \quad (26)$$

This equation is exact for incompressible fluids. The key to accurate interface advection is to realise that the discontinuous nature of the problem demands geometric modelling involving considerations of the shape and orientation of the face, as well as of the local position, orientation and motion of the interface. We formally integrate (26) over time from time t to time $t + \Delta t$:

$$\alpha_P(t + \Delta t) = \alpha_P(t) - \frac{1}{V_P} \sum_f \Delta V_f(t, \Delta t) \quad (27)$$

where $\Delta V_f(t, \Delta t)$ denotes the volume of heavy fluid transported through the face f during the time step $[t, t + \Delta t]$:

$$\Delta V_f(t, \Delta t) = \int_t^{t+\Delta t} \int_f H(\mathbf{x}, \tau) \mathbf{u}(\mathbf{x}, \tau) \cdot d\mathbf{S} d\tau. \quad (28)$$

If the flow was steady and face f completely immersed in the heavy fluid during the entire time step, this will just be $\Delta V_f(t, \Delta t) = \phi_f \Delta t$. Likewise, if the face was in the light fluid throughout the time step, $\Delta V_f(t, \Delta t)$ would be zero. But even for steady flows, some faces will in general be fully or partially swept by the interface in a non-trivial manner during a time step. In the `isoAdvector` advection step we model the face-interface intersection line sweeping the face during the time step. This approach is geometric in nature, but novel compared to existing geometric advection methods that focus on calculation of flux polyhedra and their intersection with the grid cells [21, 22, 23, 24]. The first step in our modelling process is to realise that the rapid changes in ΔV_f during a time step is typically not due to an abruptly varying velocity field but due to the passage of the interface through the cell face. Hence, we will assume that $\mathbf{u}(\mathbf{x}, \tau) \cdot d\mathbf{S}$ in (28) can be written in terms of an averaged flux over the face and over the time step:

$$\mathbf{u}(\mathbf{x}, \tau) \cdot d\mathbf{S} \approx \bar{\mathbf{u}}_f \cdot \mathbf{n}_f dA = \frac{\bar{\phi}_f}{A_f} dA, \text{ for } \mathbf{x} \in S_f \quad (29)$$

and $t \in [t, t + \Delta t]$.

Here $\bar{\mathbf{u}}_f$ and $\bar{\phi}_f$ can be thought of as averages over both time step and face area. At the beginning of the algorithm, stepping forward from time t , we may use the available $\phi_f(t)$ as the estimate of the average flux over the time step, $\bar{\phi}_f$. However, during nonlinear iterations in a single time step, the averaged flux is readily available due to the availability of $\phi_f(t + \Delta t)$. In any case, inserting (29) into (28) we can write:

$$\Delta V_f(t, \Delta t) \approx \bar{\phi}_f \int_t^{t+\Delta t} \alpha_f^+(\tau) d\tau, \quad (30)$$

where we have defined the quantity:

$$\alpha_f^+(t) = \frac{1}{A_f} \int_f H(\mathbf{x}, t) dA, \quad (31)$$

which is the instantaneous “Area-Of-Fluid” of face f , i.e. the fraction of the face area submerged in heavy fluid. If the velocity field is constant in space and time and the face is planar, the approximation in (30) becomes exact. To progress, we now assume that the interface has been reconstructed within the interface cell from which face f receives fluid (upwind cell). The reconstructed interface is represented by an internal polygonal face. We will call such a cell cutting face an *isoface*, for reasons to become clear below. The isoface cuts the cell into two disjoint sub-cells occupied by the heavy and light fluid, respectively, as illustrated in Figure 5.

The isoface will intersect some cell faces, cutting them into two subfaces immersed in heavy and light fluid, respectively, while others will be fully immersed in one of the two fluids. This is the state at time t . However, (30) requires α_f^+ for the whole interval $[t, t + \Delta t]$. To obtain an estimate of this, we first note that the isoface will have a well-defined face centre, $\mathbf{x}_S$ and a well defined unit normal, $\mathbf{n}_S$, the latter by convention pointing away from the

heavy fluid. We may then interpolate the cell averaged velocity field, $\mathbf{u}_P$ to the isoface centre, $\mathbf{x}_S$, to obtain the isoface velocity $\mathbf{u}_S$. If the fluid interface is a plane with unit normal $\mathbf{n}_S$ starting at $\mathbf{x}_S$ at time t and moving with constant velocity $\mathbf{u}_S$, then the interface will arrive at a given point $\mathbf{x}_v$ at time:

$$t_v = t + \frac{(\mathbf{x}_v - \mathbf{x}_S) \cdot \mathbf{n}_S}{\mathbf{u}_S \cdot \mathbf{n}_S}. \quad (32)$$

In particular, this holds true for all points on the general polygonal (N -sided) face f , including its vertices $\mathbf{x}_1, \dots, \mathbf{x}_N$, and therefore defines the face-interface intersection line at any $\tau \in [t, t + \Delta t]$ as required in (30). We will now use this to explicitly calculate the time integral in (30).

First note that a planar polygonal face may be triangulated in a number of ways, with the triangles lying exactly on the surface of the face. For a non-planar polygonal face we must define its surface, which we do by estimating a face centre and using that as the apex for N triangles with the N face edges as base lines. The face surface is then defined by the union of these N triangles. In other words, any polygonal face may be represented as a union of triangles. Our analysis can therefore be confined to a

Figure 5: Reconstructed “isoface” in a polyhedral interface cell.

Figure 6: Triangular subface cut by planar isoface at face-interface intersection line.

triangular subface since the contribution from these can subsequently be accumulated to obtain the time integral in (30) for the whole face. Therefore, we consider a triangle with vertices $\mathbf{x}_1, \mathbf{x}_2$ and $\mathbf{x}_3$. The interface arrival times from (32) can be calculated and we may assume without loss of generality that the points are ordered such that $t_1 \leq t_2 \leq t_3$. The interface enters the triangle at time t_1 at the point $\mathbf{x}_1$, and then sweeps the triangle reaching $\mathbf{x}_2$ at time t_2 , where it also intersects the edge $\mathbf{x}_1 - \mathbf{x}_3$ at a point we shall call $\mathbf{x}_4$, illustrated in Figure 6. In what follows, we denote an edge between $\mathbf{x}_i$ and $\mathbf{x}_j$ as $\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$. Then for $\mathbf{x}_4$ we have:

$$\mathbf{x}_{41} = \frac{\mathbf{x}_{21} \cdot \mathbf{n}_S}{\mathbf{x}_{31} \cdot \mathbf{n}_S} \mathbf{x}_{31}. \quad (33)$$

Finally, at time t_3 , the interface leaves the face through $\mathbf{x}_3$. We note that in general the three times t_1, t_2 and t_3 and the two times t and $t + \Delta t$ can be distributed in various ways. For instance if $t < t_1 < t_2 < t + \Delta t < t_3$, then the triangle is completely immersed in the light fluid from time t to time t_1 at which point the isoface will enter the triangle sweeping it and ending up at time $t + \Delta t$ on the triangle. The correct ordering must be taken into account, when doing the time integration in (30). Let us for the sake of simplicity consider the case where the triangle is entirely swept during the time step, i.e. where $t < t_1$ and $t_3 < t + \Delta t$. We will derive an expression for $\alpha^+(\tau)$ under the assumption that $\mathbf{U}_s \cdot \mathbf{n}_S > 0$, meaning that the interface is moving towards the light fluid region within the cell. If this is not the case, what we have derived is instead an expression for $1 - \alpha^+$, which is equally useful. At a time τ between t_1 and t_2 , the immersed part of the triangle will have area:

$$A^+(\tau) = \frac{1}{2} |\mathbf{x}_{41} \tilde{t} \times \mathbf{x}_{21} \tilde{t}|, \text{ where } \tilde{t} = \frac{\tau - t_1}{t_2 - t_1}. \quad (34)$$

With a total area of the triangle of $A = \frac{1}{2} |\mathbf{x}_{31} \times \mathbf{x}_{21}|$, we can then write:

$$\alpha^+(\tau) = \frac{|\mathbf{x}_{41} \times \mathbf{x}_{21}|}{2A} \left(\frac{\tau - t_1}{t_2 - t_1} \right)^2, \text{ for } t_1 < \tau < t_2. \quad (35)$$

In a similar manner we find:

$$\alpha^+(\tau) = \alpha^+(t_2) + \frac{|\mathbf{x}_{43} \times \mathbf{x}_{23}|}{2A} \left[1 - \left(1 - \frac{\tau - t_3}{t_2 - t_3} \right)^2 \right] \quad (36)$$

for $t_2 < \tau < t_3$.

From (35) and (36) it is evident that α^+ for the sub-triangles of a polygonal face are quadratic polynomials in τ with coefficients changing at the intermediate time t_2 . The coefficients are uniquely determined by the face vertex positions, $\mathbf{x}_1, \mathbf{x}_2$ and $\mathbf{x}_3$, the isoface velocity, $\mathbf{u}_s$, the unit normal, $\mathbf{n}_S$, and the isoface centre at the beginning of the time step, $\mathbf{x}_S$. In Figure 7 and Figure 8, we show an example of the time evolution of $\alpha_f^+(t)$ for a polygonal face as it is swept by a planar interface. If we name the polynomial coefficients for the first sub time interval of an polygon's i 'th triangle $A_{i,1}, B_{i,1}$ and $C_{i,1}$ (see (35)), and the coefficients for its second sub interval $A_{i,2}, B_{i,2}$

and $C_{i,2}$ (see (36)), then the time integral in (30) takes the form:

$$\int_t^{t+\Delta t} \alpha_f^+(\tau) d\tau \approx \sum_{i=1}^N \sum_{j=1}^2 \frac{1}{3} A_{i,j} (t_{i,j+1}^3 - t_{i,j}^3) + \frac{1}{2} B_{i,j} (t_{i,j+1}^2 - t_{i,j}^2) + C_{i,j} (t_{i,j+1} - t_{i,j}). \quad (37)$$

Here $t_{i,1}, t_{i,2}$ and $t_{i,3}$ are the arrival times for the i 'th triangle of our polygonal face (see (32)). The approximation in (37) is exact if the interface is in fact a plane with normal $\mathbf{n}_S$ starting at position $\mathbf{x}_S$ at time t and travelling with constant velocity $\mathbf{u}_S \cdot \mathbf{n}_S$ normal to itself.

This concludes our description of the `isoAdvect` advection step. We will now briefly describe the `isoAdvect` reconstruction step giving rise to the first syllable, the "iso", in the method name. The reconstruction step is used to obtain the isoface at the beginning of a time step including its centre $\mathbf{x}_S$ and unit normal, $\mathbf{n}_S$. As suggested by the name, this is done by representing the isoface

Figure 7: Face-interface intersection line sweeping a polygonal face and passing by its vertices.

Figure 8: The evolution of the area-of-fluid as the face is swept. Quadratic dependency on τ with different coefficients on each subinterval.

as the intersection between the cell and a numerically calculated isosurface of the volume fraction field, $\alpha_P(t)$. To calculate such isosurface, the volume fraction field is first interpolated from the cell centres to the vertices of the cell. In the current implementation the inverse centre-to-vertex distances are used as interpolation weights. With a volume fraction value associated with each cell vertex, we can now for a given iso-value, α_0 , determine for each cell edge, if α_0 lies between the two vertex values of that edge. If this is the case, we mark a cut point on the edge by linear interpolation. Doing this for all the cell's edges and connecting the cut points across the cell faces, we obtain the isoface. Its centre and normal can be calculated by triangulation as for any other polygonal face.

It is important to choose for each interface cell a distinct iso-value giving rise to an isoface cutting the cell into sub-cells of volumetric proportions in accordance with the volume fraction of the cell. The search algorithm for finding the iso-value to within a user specified tolerance has been optimized by exploiting the known functional form of a sub-cell volume as a function of the iso-value. For more details, the reader is referred to Roenby *et al.* [20].

The final element in the `isoAdvector` algorithm is a heuristic bounding step. It is introduced to correct volume fractions ending up outside the meaningful interval, $[0, 1]$, if the `isoAdvector` algorithm is stressed beyond its formal region of validity by taking time steps so large that the underlying geometric assumptions break down. The bounding step is optional and contains both a volume-preserving step and an optional non-conservative brute force chopping of the volume fractions, which is not used in this work. For more details, the reader is referred to Roenby *et al.* [20].

POLYHEDRAL ADAPTIVE GRID REFINEMENT Polyhedral Adaptive Grid Refinement (AGR) is implemented in `foam-extend-4.1` and used in this work. The first step of the AGR algorithm is to select the refinement and unrefinement candidates. The refinement candidates are selected based on the GFM data, namely all the cells sharing an interface face in the grid are marked for refinement (see Figure 4). An additional layer of face-neighbouring cells is marked for refinement starting from the initial set of interface cells. Such procedure ensures that the refined region is close to the interface where discontinuities occur and where the curvature needs to be calculated for surface tension force. Refinement and unrefinement frequency can be specified separately, although in this work, both refinement and unrefinement are performed simultaneously in each time-step. The refinement works for arbitrary polyhedral cells by adding points at the: i) Cell centre, ii) Face centres and iii) Edge centres. Each N -sided face is then split into N new faces by connecting existing corner point, edge mid-point, face centre and another edge mid-point sharing the initial corner point. The new internal faces are created by connecting the edge

mid-point, face centre, cell centre and another face centre sharing the initial edge. The method is well-defined for an arbitrary polyhedra, where the refinement for a polyhedral cell with eight faces is presented in Figure 9. Note that the refinement procedure for a hexahedral cell reduces to standard hexahedral refinement that breaks down the hexahedral cell into eight smaller hexahedral cells. Additional details regarding the implementation of the AGR in `foam-extend-4.1` are beyond the scope of this work.

Figure 9: Polyhedral cell refinement. Top to bottom: original cell, first refinement level, second refinement level.

DYNAMIC LOAD BALANCING The numerical model relies on the domain decomposition in order to ensure efficient parallelisation of the solution algorithm. At the beginning of the simulation, the domain is decomposed in N subdomains, where N is the number of processors. During the simulation, as the spray undergoes atomization and the AGR keeps the specified refinement level close to the interface, a large difference in number of cells per processor can be obtained, which significantly deteriorates the parallel performance. In order to avoid this issue, a Dynamic Load Balancing (DLB) is performed where the cells on each processor are re-distributed to other processor if a user-specified criterion is reached. The criterion is based on the number of cells per processor, where N_{min} represents the smallest number of cells on one processor and N_{max} represents the largest number of cells on another processor. If their ratio is smaller than a given input value, i.e. $N_{min}/N_{max} < \Delta$ the parallel load balancing is triggered. In this work, $\Delta = 0.7$ has been used without an attempt to optimize the overall parallel performance by changing this factor. The details regarding the implementation of the DLB in *foam-extend-4.1* are beyond the scope of this work.

SOLUTION ALGORITHM The solution algorithm based on the combination of GFM for interface discontinuity treatment, *isoAdvector* for interface advection and combined AGR and DLB for parallel performance is presented in Algorithm 1. In each time-step, the algorithm starts by performing the AGR. The DLB is performed only if the AGR is performed in the same time-step and the parallel load imbalance is detected by the Δ criterion. After AGR and DLB, a pressure correction equation is solved in order to ensure that the fluxes are solenoidal before reaching the momentum equation. The momentum equation is solved first within the nonlinear SIMPLE loop. After the momentum equation, the pressure equation is solved in an embedded PISO loop N_{PISO} times. After the momentum predictor and pressure corrector steps are finished, the interface is advected using the *isoAdvector* algorithm, providing the new volume fraction field that is then used to calculate the new GFM discretisation data for discretisation of the pressure gradient in the momentum equation and the pressure laplacian in the pressure equation. Due to the nonlinear convection term, the flow solution is repeated N_{SIMPLE} times. In this work, $N_{SIMPLE} = N_{PISO} = 2$ is used, resulting in two momentum equation solutions, two interface advection steps and four pressure equation solutions. No attempt in this work has been made to optimize the number of nonlinear (SIMPLE) iterations and pressure solution (PISO) iterations.

PRELIMINARY SIMULATION OF THE SPRAY D INJECTOR

Spray D injector nozzle #209133 in cold condition [25] is considered in order to perform a preliminary test of the developed framework. The nominal orifice diameter is 180 μm and the injection pressure is 150 MPa. The temper-

Algorithm 1 Solution algorithm for each time step.

```

Perform AGR
if AGR performed and  $N_{min}/N_{max} < \Delta$  then
    Perform DLB
end if
Solve pressure correction equation
while  $i_{SIMPLE} < N_{SIMPLE}$  do
    Solve momentum equation
    while  $i_{PISO} < N_{PISO}$  do
        Solve pressure equation
    end while
    Advect interface
    Assemble GFM discretisation data
end while

```

ature of the n-dodecane and of the ambient air is 298 K, with the ambient density of 22.8 kg/m³.

The computational grid at the beginning of the simulation is presented in ???. The grid consists of two parts: i) Cylindrical part covering a part of the nozzle that is 0.5 mm long and has a diameter of 0.191 mm; ii) Cuboid part that is 8 mm long and has a cross section of 0.8 $\times$ 0.8 mm. The base cell size is $\Delta_x = \Delta_y = \Delta_z = 40 \mu\text{m}$ in the cuboid region. The simulation starts with a small amount of fuel in the nozzle in order to initially refine the interface up to four refinement levels. This is presented in Figure 11, where the side view of the nozzle part is presented, and in Figure 12 where the slices through the nozzle grid are presented. The cells in the nozzle are initially refined two times in order to capture the jet contraction due to boundary layer effects. Two additional refinement levels are used near the initial location of the fuel-air interface, leading to the cell size of 2.5 μm . This is done in order to keep the interface Courant number nearly constant during the whole simulation. The grid initially has approximately 0.67 million cells, mostly hexahedral (98% of all cells).

The time step in the simulation is controlled by the maximum Courant number of 0.9, resulting in approximate time step of $\Delta T \approx 1.25 \text{ ns}$. The simulation has been performed in parallel on 112 cores (Intel Xeon CPU E5-2637 v3 @ 3.5 GHz). The AGR caused the grid to refine from 0.67 million cells at the beginning to 137 million cells at the end of the simulation. The simulation has been performed up to $t \approx 10.4 \mu\text{s}$, taking approximately 11 days or 30 000 core hours. In total, approximately 8000 time-steps have been performed. At the onset of the atomization, a mushroom shaped pattern is observed as can be seen in Figure 13. Figure 13 presents an iso-surface of $\alpha = 0.5$ coloured by the velocity magnitude. A similar shape has been observed both numerically and experi-

Figure 10: Side view of the whole computational grid.

mentally by Ghiji *et al.* [5]. The perspective view of the primary atomization processes is presented in Figure 14. At $t = 9 \mu\text{s}$, the jet is almost fully atomized at the distance of approximately 20 orifice diameters. Additionally, Figure 15 presents the side view of the atomization process at the same time instants.

Following Herbert *et al.* [26], a droplet identification method has been implemented in the existing framework. The method uses volume fraction field to determine connected volumes of the atomized fuel. The diameter of the connected volume is calculated based on the enclosed volume, assuming spherical shape. The method therefore does not discern different structures such as ligaments from close to spherical droplets, which represents a limitation that should be addressed in future work. The method has been used to collect the droplets from $1 \mu\text{m}$ to $20 \mu\text{m}$ in diameter far from the core in a hollow cylinder starting from 0.2 mm radially. The number of droplets far from

the core grows from approximately 3 000 at $t = 2 \mu\text{s}$ to 5 000 000 droplets at $t = 10 \mu\text{s}$, which can be seen at Figure 16. The mean droplet diameter slightly increases from $6.5 \mu\text{m}$ to $6.7 \mu\text{m}$ at the end, which is not intuitive. Furthermore, compared to experimentally measured results, the droplet diameter of approximately $d \approx 6.7 \mu\text{m}$ is significantly higher than the experimentally measured Sauter Mean Diameter (SMD) of approximately $d_{32} \approx 1.8 \mu\text{m}$ [7, 25]. The reason for such a discrepancy could lie either in the droplet detection algorithm used for post processing of the results or the computational set-up. The authors are working on finding the cause of this discrepancy before performing more detailed analysis of the results. Figure 18 presents the time evolution of the standard deviation of the droplet diameter, which is lowered from $4.3 \mu\text{m}$ at the beginning of the atomization to approximately $4 \mu\text{m}$ at the end of the simulation.

Figure 11: Zoomed side view of the whole computational grid near the nozzle.

Figure 12: Two orthogonal slices through the grid near the nozzle.

Figure 13: Side view of the atomization onset at the two time steps.

Figure 14: Perspective view of the primary atomization.

Figure 15: Side view of the primary atomization.

Figure 16: Time evolution of the number of droplets in fully atomized region.

Figure 17: Time evolution of the mean droplet diameter in fully atomized region.

Figure 18: Time evolution of the standard deviation of the droplet diameter in fully atomized region.

CONCLUSION AND DISCUSSION

The paper presents a development of the Finite Volume numerical framework implemented in *foam-extend-4.1* for the primary atomization based on the *isoAdvector*–

GFM–AGR–DLB. The methodology for interface advection and handling of discontinuities proved to be accurate in previous research [14, 20], where their combination with AGR and DLB is implemented and investigated in this work. The combined AGR and DLB enables us to run large-scale computations in an efficient manner, keeping the fine cell resolution in the area of interest near the interface using the AGR, while preventing the deterioration of parallel scalability using the DLB.

The framework has been initially tested on Spray D injector, which presents an ongoing effort. The first simulation lasted approximately 11 days on 112 cores (equivalent to 30 000 core hours) for approximately 10 μs . Although rich information on the initial transient can be obtained from such a simulation, further work needs to be done in order to validate the methodology. The discrepancy between the mean diameter of $d \approx 6.7 \mu\text{m}$ compared to the experimentally measured SMD of $d_{32} \approx 1.8 \mu\text{m}$ in the atomized region needs to be further investigated. The difference in results may be due to: i) An error in the post processing; ii) The simulated conditions; iii) The insufficient grid resolution of 2 μm near the interface (as similarly observed by Ghiji *et al.* [5] or; iv) Too short simulation that did not achieve quasi steady-state conditions. All of these factors are currently under investigation and we hope to find the cause of the discrepancy and publish it in future publications.

ACKNOWLEDGEMENTS

This research was sponsored by the King Abdullah University of Science and Technology, Saudi Arabia within the Competitive Research Grant: Predictive Models and Experimental Validation of Multicomponent Dense Spray Dynamics, OSR-2017-CRG6-3409.03, under the administration of Prof. Hong Im.

REFERENCES

- [1] S. Subramaniam, Lagrangian–Eulerian Methods for Multiphase Flows, *Progress in Energy and Combustion Science* 215 (2013) 215–245.
- [2] J. K. Dukowicz, A Particle-Fluid Numerical Method for Liquid Sprays, *Journal of Computational Physics* 35 (2) (1980) 229–253.
- [3] Petranović, Z. and Edelbauer, W. and Vujanović, M. and Duić, N., Modelling of the reactive sprays employing the Euler Eulerian multi-continuum approach, in: *Proceedings of the 27th European Conference on Liquid Atomization and Spray Systems (ILASS)*, Brighton, United Kingdom, 2016.
- [4] O. Desjardins, V. Moureau, H. Pitsch, An accurate conservative level set/ghost fluid method for simulating turbulent atomization, *J. Comput. Phys.* 227 (18) (2008) 8395–8416.
- [5] M. Ghiji, L. Goldsworthy, P. A. Brandner, V. Garaniya, P. Hield, Numerical and experimental investigation of

- early stage diesel sprays, *Fuel* 175 (2016) 274–286. doi:10.1016/j.fuel/2016.02.040.
- [6] M. Arienti, M. Sussman, A numerical study of the thermal transient in high-pressure diesel injection, *International Journal of Multiphase Flow* 88 (2017) 205–221. doi:10.1016/j.ijmultiphaseflow.2016.09.017.
- [7] M. Battistoni, G. M. Magnotti, C. L. Genzale, M. Arienti, K. E. Matusik, D. J. Duke, J. Giraldo, J. Ilavsky, A. L. Kastengren, C. F. Powell, P. Marti-Aldavari, Experimental and Computational Investigation of Sub-critical Near-Nozzle Spray Structure and Primary Atomization in the Engine Combustion Network Spray D, SAE Technical Paper 2018-01-0277 doi:doi:10.4271/2018-01-0277.
- [8] M. Jemison, M. Sussman, M. Arienti, Compressible, multiphase semi-implicit method with moment of fluid interface representation, *J. Comput. Phys.* 279 (2014) 182–217.
- [9] V. Vukčević, Numerical modelling of coupled potential and viscous flow for marine applications, Ph.D. thesis, Faculty of Mechanical Engineering and Naval Architecture, University of Zagreb, PhD Thesis (2016). doi:10.13140/RG.2.2.23080.57605.
- [10] J. Huang, P. M. Carrica, F. Stern, Coupled ghost fluid/two-phase level set method for curvilinear body-fitted grids, *Int. J. Numer. Meth. Fluids* 44 (2007) 867–897. doi:10.1002/fld.1499.
- [11] H. Jasak, Error analysis and estimation for the finite volume method with applications to fluid flows, Ph.D. thesis, Imperial College of Science, Technology & Medicine, London (1996).
- [12] S. V. Patankar, D. B. Spalding, A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows, *Int. J. Heat Mass Transf.* 15 (1972) 1787–1806.
- [13] R. I. Issa, Solution of the implicitly discretised fluid flow equations by operator-splitting, *J. Comput. Phys.* 62 (1986) 40–65.
- [14] V. Vukčević, H. Jasak, I. Gatin, Implementation of the Ghost Fluid Method for Free Surface Flows in Polyhedral Finite Volume Framework, *Comput. Fluids* 153 (2017) 1–19. doi:10.1016/j.compfluid.2017.05.003.
- [15] C. M. Rhie, W. L. Chow, A numerical study of the turbulent flow past an isolated airfoil with trailing edge separation, *AIAA J.* 21 (1983) 1525–1532.
- [16] J. H. Ferziger, M. Peric, *Computational Methods for Fluid Dynamics*, Springer, 1996.
- [17] H. Jasak, H. G. Weller, Application of the finite volume method and unstructured meshes to linear elasticity, *Int. J. Numer. Methods Eng.* 48 (2000) 267–287.
- [18] C. Ivey, P. Moin, Accurate interface normal and curvature estimates on three-dimensional unstructured non-convex polyhedral meshes, *J. Comp. Phys.* 300 (2014) 365–386. doi:10.1016/j.jcp.2015.07.055.
- [19] F. Denner, B. G. Van Wachem, Fully-coupled balanced-force VOF framework for arbitrary meshes with least-squares curvature evaluation from volume fractions, *Numerical Heat Transfer, Part B: Fundamentals* 65 (3) (2014) 218–255. arXiv:1405.0829, doi:10.1080/10407790.2013.849996.
- [20] J. Røenby, H. Bredmose, H. Jasak, A computational method for sharp interface advection, *Open Science* 3 (11). doi:10.1098/rsos.160405.
- [21] B. Xie, S. Li, F. Xiao, An efficient and accurate algebraic interface capturing method for unstructured grids in 2 and 3 dimensions: The THINC method with quadratic surface representation, *International Journal for Numerical Methods in Fluids* 76 (12) (2014) 1025–1042. arXiv:fld.3968, doi:10.1002/fld.3968.
- [22] J. López, C. Zanzi, P. Gómez, F. Faura, J. Hernández, A new volume of fluid method in three dimensions - Part II: Piecewise-planar interface reconstruction with cubic-Bézier fit, *International Journal for Numerical Methods in Fluids* 58 (8) (2008) 923–944. arXiv:fld.1, doi:10.1002/fld.1775.
- [23] J. Hernández, J. López, P. Gómez, C. Zanzi, F. Faura, A new volume of fluid method in three dimensions - Part I: Multidimensional advection method with face-matched flux polyhedra, *International Journal for Numerical Methods in Fluids* 58 (8) (2008) 897–921. arXiv:fld.1, doi:10.1002/fld.1776.
- [24] H. T. Ahn, M. Shashkov, Multi-material interface reconstruction on generalized polyhedral meshes, *Journal of Computational Physics* 226 (2) (2007) 2096–2132. doi:10.1016/j.jcp.2007.06.033.
- [25] Engine Combustion Network, Engine Combustion Network Experimental Data Archive, <https://ecn.sandia.gov>, [Online; accessed 21 March 2018] (2018).
- [26] D. A. Herbert, D. P. Schmidt, D. A. Knaus, S. Phillips, P. J. Magari, Parallel VOF Spray Droplet Identification in an Unstructured Grid, in: *Proceedings of the 21st Annual Conference on Liquid Atomization and Spray Systems (ILASS)*, Orlando, Florida, 2008.