Finite Pointset Method – a numerical method for simulations in gas dynamics. Special applications to airbag deployment simulations.

Dr. Jörg Kuhnert
Group leader meshfree methods
Fraunhofer Institut Techno- und Wirtschaftsmathematik (ITWM)
Gottlieb-Daimler-Str., Geb 49
67663 Kaiserslautern
GERMANY
email: kuhnert@itwm.fraunhofer.de
phone: ++49 / 631 / 205 4087
fax: ++49 / 631 /205 4139

Abstract: *In the past years, the so-called Finite Pointset Method (FPM), a new meshfree numerical method for solving problems in gas- and hydrodynamics, has been developed at ITWM in Kaiserslautern. The main ideas of FPM are as follows.*
- Fluid flows, governed by the compressible Euler equations, are modeled.
- The flow domain is filled with a cloud of numerical points (also called particles). Particles are carriers of all information relevant to the considered physical problem.
- There is no meshing of the particles. Instead, particles in a ball around some point are called neighbors. The radius of the ball (smoothing length) is user given or adaptive, and may change locally.
- Derivatives, used to approximate the Euler equations, are modeled by the moving least squares (MLS) operator.
- The scheme is based on a Lagrange idea, i.e. the particles are moving with fluid velocity. Therefore, an excellent adaptivity to dynamic geometries or free surfaces is given.
- It will be shown that the scheme is locally and continuously conservative.
- An upwind idea is presented which is essential to stabilize the scheme.
By its character, FPM is a general finite difference approach.
FPM was successfully employed already in several industrial projects. Special effort was done in the field of airbag deployment simulations in case of car accidents. Here, FPM was integrated into the commercial PAM-CRASH software, developed by ESI-Group.

Keywords: *Meshfree Methods, Finite Pointset Method, General Finite Difference, Moving Least Squares, Upwind Techniques, Airbag Deployment Simulations, PAM-CRASH*

1. Introduction

The Finite Pointset Method (FPM) is a meshfree numerical scheme designed to solve problems in continuum mechanics. In this paper, we will focus on inviscid problems in gas dynamics (compressible fluid flows). FPM was developed at the Fraunhofer Institute for industrial mathematics (ITWM) in Kaiserslautern, Germany.

Why meshfree? Many engineers and scientists, dealing with computational fluid dynamics in its daily business, find problems, which are difficult to solve with the classical meshbased algorithms, such as Fem or FVM. There might be problems with dynamic geometries, there might be problems with very complex geometries, and there might be problems with dynamic free surfaces as well as multi phase problems with dynamic phase boundaries. All of these problems have one thing in common: In order to analyze them using meshbased methods, big effort would have to be made either for mesh generation or/and mesh maintenance. For some cases, the computational cost caused by the mesh itself dominates the cost due to the flow solver.

So, a meshfree scheme would have a big advantage: no computational cost spent on administration of meshes of any kind.

FPM requires the construction of point clouds, which are much cheaper in their handling. The points of the cloud are also referred to as particles. The point cloud represents the fluid and is subject to certain quality constraints. The most important parameters are $\mathrm{r}_{\min}$ (sphere radius about a particle, no other particle is allowed to be inside of this sphere) and $\mathrm{r}_{\max}$ (any sphere inside of the flow domain having radius $\mathrm{r}_{\max}$ has to contain at least one particle). We will frequently use the parameter h, also referred to as smoothing length or interaction radius, thus forming the sphere of influence. The parameter of h describes the region of influence about a point by $\mathbf{B}(\mathbf{y}) = \left\{ \mathbf{x} \mid \|\mathbf{y}-\mathbf{x}\|_2 \le \mathrm{h}(\mathbf{x}); \mathbf{x} \in \Omega; \mathbf{y} \in \Omega \right\}$, where $\Omega \subset \mathbf{R}^{\nu}$, $\nu = 1,2,3$ represents the flow domain. All points being located inside of this sphere are called neighbors to the particular point $\mathbf{y}$. The function h does not need to be constant throughout Ω, rather it can be a smooth function with the natural requirement $\|\nabla \mathrm{h}\| < 1$.

Using h, $\mathrm{r}_{\min}$ and $\mathrm{r}_{\max}$ can be written as $\mathrm{r}_{\max} = \alpha_{\max} \cdot \mathrm{h}$ and $\mathrm{r}_{\min} = \alpha_{\min} \cdot \mathrm{h}$, respectively. Practical values are $\alpha_{\max} = 0.45$ and $\alpha_{\min} = 0.25$

h might be a user given function, or it might be adaptive to geometrical constraints. The accuracy of the numerical computations can be measured in orders of h.

The particles are carriers of all relevant physical information required to describe the particular flow problem. In particular, the points carry information such as local density, momentum and energy. Moreover, the points move with fluid velocity, hence in each time cycle, they step "forward" in space. The particle movement might, however, affect the quality of the point cloud. Thus, regular quality checks and repair-operations (point fill-in, point-clustering) are necessary to maintain the regularity of the computation.

As we restrict ourselves to compressible, inviscid flows, the governing equations are the classical Euler equations in gas dynamics. See section 2.

In order to approximate the occurring derivatives based on point clouds, we have developed a special moving least squares (MLS) idea. The employment of MLS makes FPM a general finite difference (FD) method, which would degenerate to a classical FD method if the points would be placed in a regular meshgrid. See section 3.

The development of FPM started in the year 1996. Within a PhD-work, ITWM tried to overcome several disadvantages of the Smoothed Particle Hydrodynamics (SPH), the up-to-date meshfree method by that time. The outcome of this PhD-work was a new, upwind-based meshfree method. See section 4.

The first industrial application of FPM was carried out together with the company ESI Group in the field of airbag deployment simulations for car accident studies. The cooperation between ITWM and ESI started in 1999 and continues until today. FPM was integrated into ESI's commercial software. There, it is used to simulate the rapid gas dynamical processes of an inflating airbag. See section 5.

2. Inviscid compressible flow problems: Idea of meshfree numerical scheme

The governing equations are the Euler equations in gas dynamics, written in Lagrange form:

$$\frac{\mathrm{d}}{\mathrm{dt}}\mathrm{x} = \mathbf{v}$$ material movement

$$
\begin{aligned}
&\tfrac{d}{dt}\rho + \rho\cdot\nabla^{T}\mathbf{v} = 0 && \text{mass conservation}\\
(1)\qquad &\tfrac{d}{dt}(\rho\mathbf{v}) + \rho\mathbf{v}\cdot\nabla^{T}\mathrm{v} = -\nabla p && \text{momentum conservation}\\
&\tfrac{d}{dt}(\rho E) + (\rho E)\cdot\nabla^{T}\mathbf{v} = -\nabla(p\mathbf{v}) && \text{energy conservation}\\
&p = f(\rho,\rho\mathbf{v},\rho E) && \text{equation of state}
\end{aligned}
$$

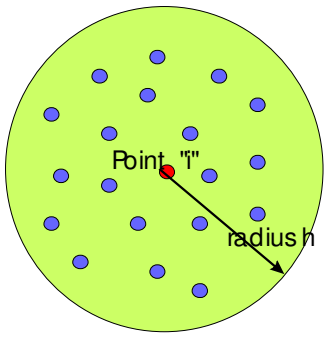
Here, the operator $\frac{d}{dt}$ denotes the change of some physical quantity along the path of a fluid particle, that means it denotes the changes an observer would feel by moving exactly with fluid velocity.

We try to use this in order to construct a Lagrangian numerical scheme. To achieve this, we establish a point cloud (see section 1). We let the points move with fluid velocity. Each point is carrier of all relevant physical information. Related to equation (1) in particular, each point at least carries the information of position, local density, momentum, and total energy. We let the points move in small time steps, and according to the model (1), the physical quantities have to be modified.

In order to directly employ the model, we have to give valuable and accurate approximations of the occurring gradients and divergences (remember that the point cloud provides only discrete function values, given at the locations of the particles). Therefore, in the next section, we explain how to model derivatives in a sufficiently accurate way.

3. Moving least squares operator for function derivatives

The Euler equations contain the gradient of pressure, the divergence of velocity as well as the divergence of the working power of the pressure. All these require the ability to approximate first spatial derivatives of function values based on point clouds.



The task consists in the approximation of derivatives of some function at any point $\mathbf{y}$ (for example at the particular point $\mathbf{x}_i$ in the picture above), just by knowing the locations of the neighbors as well as their discrete function values. Therefore, we construct a smooth, but accurate function approximating the discrete function values. We assume, that the derivatives of that smooth function approximate the original derivatives in a satisfactory way.

3.1. Procedure for MLS approximations on point clouds

Here the procedure:

1.) Construct a local polynomial of order “d” by minimizing the functional

(2)
$$\sum W\left(\frac{\left\|\mathbf{y}-\mathbf{x}_j\right\|}{h_j^2}\right)\cdot\left(f_j-p_d\left(\mathbf{y},\mathbf{x}_j\right)\right)^2 \quad \stackrel{!}{=} \quad \min$$

Here,

f_j are the discrete function values at the neighbors

$\mathbf{y}=\left(y^1 \quad y^2 \quad y^3\right)$ is the central point

$\mathbf{x}_j=\left(x_j^1 \quad x_j^2 \quad x_j^3\right)$ are the locations of the neighbors to $\mathbf{y}$

$W\left(\frac{\left\|\mathbf{y}-\mathbf{x}_j\right\|}{h_j^2}\right)$ is a weight function, which is smooth and which is zero outside $\mathbf{B}(\mathbf{y})$

$p_d(\mathbf{y},*)$ is the polynomial to be constructed around the point $\mathbf{y}$, which has the form
$p_d\left(\mathbf{y},\mathbf{x}_j\right)= c_0 + c_1\cdot\left(y^1 - x_j^1\right) + c_2\cdot\left(y^2 - x_j^2\right) + c_3\cdot\left(y^3 - x_j^3\right) + \ ...$

2.) construct a **global** smooth function by using the **local** polynomial approximations

(3)
$$\Pi f(\mathbf{y}) \equiv p_d(\mathbf{y},\mathbf{y}) \qquad \Rightarrow \qquad \Pi f(\mathbf{y}) = c_0(\mathbf{y})$$

One can prove that the function Πf is, globally, at least as smooth as the weight function W.

3.) Compute the derivatives $\frac{\partial \Pi f}{\partial y^k}$ analytically, which is possible as we will see in the next paragraph 3.2.

3.2. Numerical implementation and Shape functions of the MLS approximation

The procedure introduced in formula (2) leaves the question, how this is set into practice. We will answer to this. The system (2) can be rewritten as an over determined linear system, to be solved with respect to a least squares constraint.

(4)
$$\left\|\mathbf{W}\cdot\mathbf{A}\cdot\mathbf{c}-\mathbf{W}\cdot\mathbf{f}\right\|_2 \stackrel{!}{=} \min$$

where we have

$\mathbf{c}(\mathbf{y})=\left(c_0 \quad c_1 \quad c_2 \quad c_3 \quad \ldots\right)^T$ the vector of the polynomial coefficients

$\mathbf{f}(\mathbf{y})=\left(f_1 \quad \ldots \quad f_N\right)^T$ the vector of discrete function values of the neighbors of $\mathbf{y}$

$$\mathbf{A}(\mathbf{y})=\begin{pmatrix} 1 & \left(y^1-x_1^1\right) & \left(y^2-x_1^2\right) & \left(y^3-x_1^3\right) & \cdots \\ \vdots & \vdots & \vdots & \vdots & \cdots \\ 1 & \left(y^1-x_N^1\right) & \left(y^2-x_N^2\right) & \left(y^3-x_N^3\right) & \cdots \end{pmatrix}$$ the matrix of monomials of the neighbors

$$\mathbf{W}(\mathbf{y}) = \begin{pmatrix} W(\mathbf{y},\mathbf{x}_1) & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & W(\mathbf{y},\mathbf{x}_N) \end{pmatrix}$$ (diagonal) matrix of the neighbor weights

In order to find the polynomial, required by (2), we have to solve for the vector $\mathbf{c}(\mathbf{y})$ in the linear least squares system (4). The solution is given by

(5) $$\mathbf{c(y)} = [\,(\mathbf{WA})^{\mathbf{T}} \cdot \mathbf{WA}\,]^{-1} \cdot [\,(\mathbf{WA})^{\mathbf{T}} \cdot \mathbf{W}\,] \cdot \mathbf{f}$$

By definition (3) we need to find the derivatives of $\mathbf{c(y)}$ (strictly spoken only the derivatives of its first element). However the derivatives of $\mathbf{c(y)}$ can be computed analytically as

(6) $$\frac{\partial}{\partial \mathbf{y}^k}\mathbf{c} = [\mathbf{WA}^{\mathbf{T}} \cdot \mathbf{WA}]^{-1} \cdot \left(\frac{\partial}{\partial \mathbf{y}^k}[\mathbf{WA}^{\mathbf{T}} \cdot \mathbf{W}] - \frac{\partial}{\partial \mathbf{y}^k}[\mathbf{WA}^{\mathbf{T}} \cdot \mathbf{WA}] \cdot [\mathbf{WA}^{\mathbf{T}} \cdot \mathbf{WA}]^{-1} \cdot \mathbf{W} \right) \cdot \mathbf{f}$$

The computation of $\frac{\partial}{\partial \mathbf{y}^k}\mathbf{c}$ only requires the knowledge of the analytical derivatives of $\mathbf{W}$ and $\mathbf{A}$, however these are easy to be found since $\mathbf{A}$ consists of simple monomials, and $\mathbf{W}$ consists of the weight function, which is assumed to be smooth and, indeed, given by the user.

An interesting aspect is the fact, that the approximation of a function with respect to the point cloud (equation (5)), is naturally split into a geometrical part and a function value part. The geometrical part is the term $[\,(\mathbf{WA})^{\mathbf{T}} \cdot \mathbf{WA}\,]^{-1} \cdot [\,(\mathbf{WA})^{\mathbf{T}} \cdot \mathbf{W}\,]$, being dependent only on the location of the neighbor points. We define the j-th column of this term to be the shape function at the location $\mathbf{y}$ around the point $\mathbf{x}_j$ in the sense

(7) $$\mathbf{\Psi}(\mathbf{y},\mathbf{x_j}) \equiv \mathrm{j-th\ column\ of}\ [\,(\mathbf{WA})^{\mathbf{T}} \cdot \mathbf{WA}\,]^{-1} \cdot [\,(\mathbf{WA})^{\mathbf{T}} \cdot \mathbf{W}\,]$$

and so we can rewrite (5) by the scalar product

(8) $$\mathbf{\Pi f}(\mathbf{y}) = \mathbf{c}(\mathbf{y}) = \sum_{j=1}^{N} \mathbf{\Psi}(\mathbf{y},\mathbf{x_j}) \cdot \mathrm{f_j}$$

and consequently, (6) can be rewritten by the scalar product

(9) $$\frac{\partial}{\partial \mathbf{y}^k}\mathbf{\Pi f}(\mathbf{y}) = \frac{\partial}{\partial \mathbf{y}^k}\mathbf{c}(\mathbf{y}) = \sum_{j=1}^{N}\left[\frac{\partial}{\partial \mathbf{y}^k}\mathbf{\Psi}(\mathbf{y},\mathbf{x_j})\right] \cdot \mathrm{f_j}$$

The results of (8) and (9) have two major consequences. On one hand, we can store the discrete values of $\mathbf{\Psi}(\mathbf{x_i},\mathbf{x_j})$ as well as $\frac{\partial}{\partial \mathbf{y}^k}\mathbf{\Psi}(\mathbf{x_i},\mathbf{x_j})$, if they are nonzero (i.e. if $\mathbf{x_i},\mathbf{x_j}$ is a pair of neighbors) and by this we are able to efficiently evaluate the approximations of many functions and their derivatives in a single loop. On the other hand, $\frac{\partial}{\partial \mathbf{y}^k}\mathbf{\Psi}(\mathbf{x_i},\mathbf{x_j})$ will have to degenerate to the classical finite difference operators, if the neighbors are arranged in a regular cubic sense. Therefore, we call FPM, which is based on the MLS presented here, a generalized finite difference scheme.

3.3. Local accuracy of the MLS

The local accuracy of the approximations produced by the presented MLS depends strongly on the order of the local polynomial $p_d(\mathbf{y}, \mathbf{x}_j)$. For smooth (i.e. analytical) functions f, one can prove, that there is a constant such that

$$\left| \frac{\partial}{\partial \mathbf{y}^k} \mathbf{\Pi} f(\mathbf{y}) - \frac{\partial}{\partial \mathbf{y}^k} f(\mathbf{y}) \right| \leq K_d \cdot h^d(\mathbf{y})$$

where K_d is bounded (only if f is not smooth enough K_d might become large). So, the accuracy of the approximation is of order d, provided that $p_d(\mathbf{y}, \mathbf{x}_j)$ is of the same order.

4. Meshfree numerical upwind scheme

4.1. Central, but unstable scheme

For numerical integration of the Euler equations, the system of differential equations (1) could be rewritten such, that the occurring derivatives are modeled through the MLS approximation. Hence, the system (1) has the following numerical form

$$\begin{aligned}
p_i &= f(\rho_i, \rho \mathbf{v}_i, \rho E_i) \\
\tfrac{d}{dt}(\rho_i) &= -\rho_i \cdot (\nabla \Pi \mathbf{v}^T)_i \\
\tfrac{d}{dt}(\rho \mathbf{v}_i) &= -(\rho \mathbf{v}_i) \cdot (\nabla \Pi \mathbf{v}^T)_i - (\nabla \Pi p)_i \\
\tfrac{d}{dt}(\rho E_i) &= -(\rho E_i) \cdot (\nabla \Pi \mathbf{v}^T)_i - \nabla(\Pi p \cdot \Pi \mathbf{v}^T)_i \\
\tfrac{d}{dt} x_i &= \Pi \mathbf{v}_i
\end{aligned} \qquad \textbf{(10)}$$

Here, the "i" is a counting index for the particles. As the system (1) is, in principle, a system of hyperbolic equations (eigenvalues of $(0 \quad c \quad -c)$, c represents the sound speed induced by the equation of state $c^2 = \left.\frac{\partial p}{\partial \rho}\right|_s$), we find that the numerical scheme (10) will behave in an unstable way. Why this? The scheme presented above will reduce to a central finite difference scheme in case the point cloud is distributed in a regular way. Thus, one can say that the scheme above is a general centralized finite difference scheme. However, central finite differences for hyperbolic laws lead to numerical instabilities.

4.2. Upwind idea by splitting the scheme

A very powerful method in order to stabilize the scheme is an upwinding technique. Here, we define an upwind direction $\mathbf{n}$ and perpendicular directions $\mathbf{m}$ and $\mathbf{q}$ and rewrite the system (10) in form of two subsystems with the help of these new coordinate directions.

$$p_i = f(\rho_i, \rho \mathbf{v}_i, \rho E_i)$$

$$\frac{d}{dt}(\rho_i) = -\rho_i \cdot \left(\frac{\partial \mathbf{v}_i^{\,n}}{\partial n}\right)_i \qquad -\rho_i \cdot \left(\frac{\partial \mathbf{v}_i^{\,m}}{\partial m} + \frac{\partial \mathbf{v}_i^{\,q}}{\partial q}\right)_i$$

(11)
$$\frac{d}{dt}(\rho \mathbf{v}_i) = -(\rho \mathbf{v}_i) \cdot \left(\frac{\partial \mathbf{v}^{n}}{\partial n}\right)_i - \left(\frac{\partial p^{n}}{\partial n} \cdot \mathbf{n}\right)_i \qquad -(\rho \mathbf{v}_i) \cdot \left(\frac{\partial \mathbf{v}^{m}}{\partial m} + \frac{\partial \mathbf{v}^{q}}{\partial q}\right)_i - \left(\frac{\partial p^{m}}{\partial m} \cdot \mathbf{m} + \frac{\partial p^{q}}{\partial q} \cdot \mathbf{q}\right)_i$$

$$\frac{d}{dt}(\rho E_i) = -(\rho E_i) \cdot \left(\frac{\partial \mathbf{v}^{n}}{\partial n}\right)_i - \frac{\partial}{\partial n}\left(p \cdot \mathbf{v}^{n}\right)_i \qquad -(\rho E_i) \cdot \left(\frac{\partial \mathbf{v}^{m}}{\partial m} + \frac{\partial \mathbf{v}^{q}}{\partial q}\right)_i - \left(\frac{\partial}{\partial m}\left(p \cdot \mathbf{v}^{m}\right) + \frac{\partial}{\partial q}\left(p \cdot \mathbf{v}^{q}\right)\right)_i$$

$$\frac{d}{dt} x_i = \mathbf{v}_i$$

(for simplicity, we omitted the Π as the usage of the smooth approximation operator is obviously shown in (10)).

The first subsystem is restricted to the direction of $\mathbf{n}$ only, the second subsystem is restricted to the perpendicular directions $\mathbf{m}$ and $\mathbf{q}$. For the upwinding procedure, we restrict ourselves to the direction of $\mathbf{n}$. We choose

(12)
$$\mathbf{n} = \frac{\nabla \Pi p}{\left\| \nabla \Pi p \right\|} \qquad \text{(upwind direction)}$$

as we can show that this minimizes the magnitude of the second (remaining) subsystem. The upwinding of the $\mathbf{n}$-based subsystem is easy, as this is simply a procedure in a one-dimensional (sub)space. We do not show this procedure here in detail, as it is only straight forward, but lengthy. We, of course, have to rewrite the first subsystem in its characteristic form. The characteristic form of a system of hyperbolic equations determines a set of characteristic information, each piece of which traveling with a dedicated characteristic speed in the direction of $\mathbf{n}$. One can now upwind the characteristic system by switching the evaluation of the spatial derivatives a little bit to the direction, from which characteristic information is coming from. So, we have to define an offset evaluation of the smooth approximations by

(13)
$$\Pi f^{+}(\mathbf{y}) \equiv \Pi f\left(\mathbf{y} + \alpha_{uw} \cdot h(\mathbf{y}) \cdot \mathbf{n}\right)$$
$$\Pi f^{-}(\mathbf{y}) \equiv \Pi f\left(\mathbf{y} - \alpha_{uw} \cdot h(\mathbf{y}) \cdot \mathbf{n}\right)$$

So, the evaluation of the smooth approximations is shifted by a little bit in positive and negative upwind directions. $\alpha_{uw} = 0.1...0.3$

The outcome of the upwinding procedure (first subsystem) combined with the remaining second subsystem leads to the occurrence of special upwind pressure and upwind velocity terms. With the help of these terms, the complete system can be rewritten as

$$p_i = f\left(\rho_i, \rho \mathbf{v}_i, \rho E_i\right)$$

$$\frac{d}{dt}(\rho_i) = -\rho_i \cdot (\nabla \Pi \mathbf{v}_{uw}^{T})_i$$

$$\frac{d}{dt}(\rho \mathbf{v}_i) = -(\rho \mathbf{v}_i) \cdot (\nabla \Pi \mathbf{v}_{uw}^{T})_i - (\nabla \Pi p_{uw})_i$$

$$\frac{d}{dt}(\rho E_i) = -(\rho E_i) \cdot (\nabla \Pi \mathbf{v}_{uw}^{T})_i - \nabla\left(\Pi p_{uw} \cdot \Pi \mathbf{v}_{uw}^{T}\right)_i$$

$$\tfrac{\mathrm{d}}{\mathrm{dt}}\mathrm{x}_{\mathrm{i}} = \left(\Pi \mathbf{v}_{\mathrm{uw}}\right)_i$$

Any numerical ODE solver can perform the time integration of the scheme above, as this is a system of ordinary differential equations.

The upwind velocity and pressure are a direct outcome of the upwinding procedure and have the form

$$\Pi \mathbf{v}_{\mathrm{uw}} \equiv \Pi \mathbf{v} + \Pi\left(\frac{\mathrm{p}\cdot\mathbf{n}}{2\rho \mathrm{c}}\right)^{+} - \Pi\left(\frac{\mathrm{p}\cdot\mathbf{n}}{2\rho \mathrm{c}}\right)^{-}$$

$$\Pi \mathrm{p}_{\mathrm{uw}} \equiv \Pi \mathrm{p} + \Pi\left(\frac{(\mathbf{v}^{\mathrm{T}}\mathbf{n})\cdot\rho \mathrm{c}}{2}\right)^{+} - \Pi\left(\frac{(\mathbf{v}^{\mathrm{T}}\mathbf{n})\cdot\rho \mathrm{c}}{2}\right)^{-}$$

So, $\Pi \mathbf{v}_{\mathrm{uw}}$ and $\Pi \mathrm{p}_{\mathrm{uw}}$ have the same smoothness than $\Pi \mathbf{v}$ and $\Pi \mathrm{p}$, and (important!) are in the same way analytically differentiable. The global differentiability gives rise to the investigation of local and global conservation properties of the numerical scheme of FPM.

4.3. Conservation properties of the upwind scheme

We write the numerical upwind scheme in a simpler form

(14) $$\tfrac{\mathrm{d}}{\mathrm{dt}}\mathbf{\Phi} + \mathbf{\Phi}\cdot\nabla^{\mathrm{T}}\Pi \mathbf{v}_{\mathrm{uw}} + \nabla^{\mathrm{T}}\Pi \mathbf{F}_{\mathrm{uw}} = \mathbf{S}$$

(15) $$\tfrac{\mathrm{d}}{\mathrm{dt}}\mathbf{x} = \Pi \mathbf{v}_{\mathrm{uw}}$$

where $\mathbf{\Phi} = (\rho, \rho\mathbf{v}, \rho \mathrm{E})$ represents the conservative physical quantities and $\mathbf{F}_{\mathrm{uw}}$ the appropriate flux functions, $\mathbf{S}$ will be relevant if gravity plays a role.

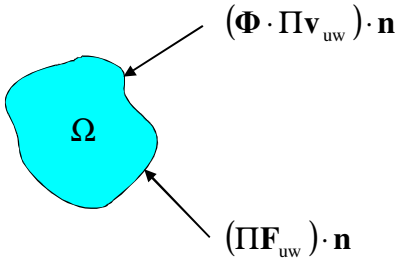
(14) together with (15) can be rewritten as

(16) $$\tfrac{\partial}{\partial \mathrm{t}}\mathbf{\Phi} + \nabla^{T}\left(\mathbf{\Phi}\cdot\Pi \mathbf{v}_{\mathrm{uw}}\right) + \nabla^{T}\Pi \mathbf{F}_{\mathrm{uw}} = \mathbf{S}$$

Now we chose an arbitrary control element Ω^{*} being a subset of the flow domain. It is easy to integrate the model (16) over the control element, and by using well-known integration rules, we find

(17) $$\frac{\mathrm{d}}{\mathrm{dt}}\int_{\Omega^{*}} \mathbf{\Phi}\, d\Omega^{*} + \oint_{\partial\Omega^{*}} \left(\mathbf{\Phi}\cdot\Pi \mathbf{v}_{\mathrm{uw}}\right)\cdot\mathbf{n}\, d(\partial\Omega^{*}) + \oint_{\partial\Omega^{*}} \left(\Pi \mathbf{F}_{\mathrm{uw}}\right)\cdot\mathbf{n}\, d(\partial\Omega^{*}) = \int_{\Omega} \mathbf{S}\, d\Omega^{*}$$

From result (17) we conclude, that the numerical contents of some physical quantity in a smooth, non-moving control volume numerically changes by the transport of this quantity through upwind velocity as well as by the upwind based flux. That is indeed true for any smooth control element.



The consequence is, that the scheme reproduces the Rankine-Hugoniot-conditions of occurring shocks in a perfect way. That means in particular, even the numerically computed speed of traveling shocks is equal to the analytical one.

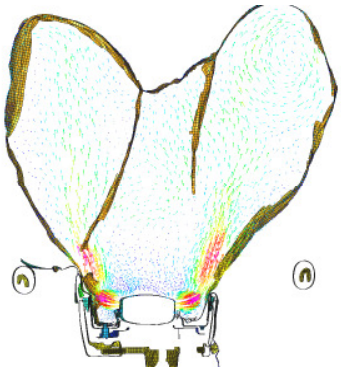
5. Applications to Airbag deployment simulations

The applicability of FPM for industrial problems was shown at first, when ITWM and ESI-Group started a co-operation in 1999. What was the background? Car as well as airbag producers were in need of reliable simulations for airbag deployment situations, as the functionality of airbags has become subject to very tight regulations. Special focus is directed to out-of-position cases. This means situations, where the driver/passenger is located very close to the unfolding airbag and therefore, in case of an accident, gets into contact with it in a very early, dangerous stage.

FPM was incorporated into PAM-CRASH, in order to simulate the gas dynamics inside of an airbag. PAM-CRASH is a well-known commercial software for crash simulations developed by ESI. FPM is now an inherent part of this software, capable handle even airbags with a complex folding topology.

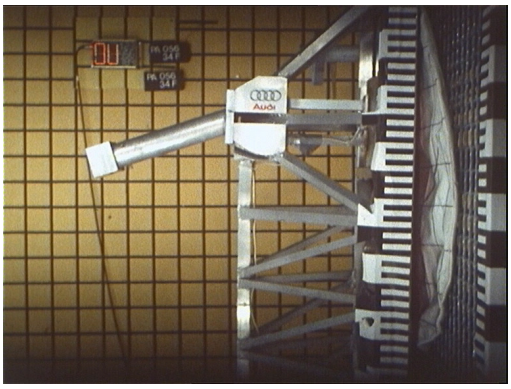
A modern car is equipped with a series of airbags (driver-, passengers-, side-impact-airbags), which are launched in a selective way in case of an accident. Out-of-Position situation (last picture). Courtesy of TRW.

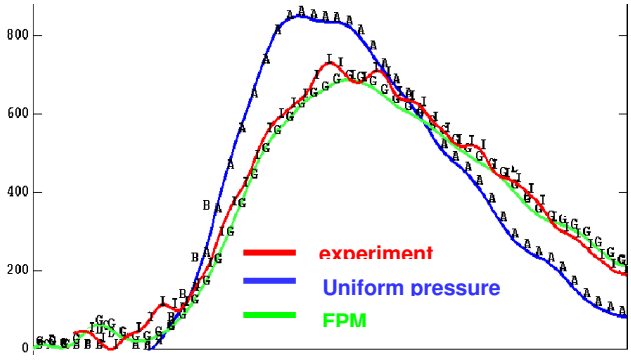
Since it is widely used for airbags, FPM has to pass several tests. One of the simple tests is the drop test or pendulum test. Here, a non-folded (preferably) or folded airbag is launched in the laboratory. The deploying airbag pushes away a pendulum or a dummy, outfitted with acceleration sensors. The measured accelerations (versus time) need to be reproduced by the numerical scheme. FPM shows a very good reliability.



Cut through an industrial airbag. Velocity distribution of the gas inside (red high, blue low), each velocity vector represents a point of the point cloud. Courtesy of TRW.

The coupling of FPM and PAM-CRASH is a strong coupling. FPM is able to handle the gas dynamics inside of an airbag, provided the knowledge of the geometrical configuration of the airbag. PAM-CRASH is able to handle the dynamics of an airbag membrane, provided the knowledge local pressure inside of the bag. The coupling therefore is an exchange of information at each time step. FPM provides the local membrane pressure to PAM-CRASH. After execution of one step, PAM-CRASH provides the new membrane position back to FPM. With this new information, FPM is able to execute the next time cycle. This procedure continues.





Experimental field: unfolded airbag against pendulum. Acceleration curves measured, with FPM, and with the classical uniform pressure model. Courtesy of Audi.

Bibliography

DILTS G. A., Moving least squares particle hydrodynamics I, consistency and stability, *Hydrodynamics methods group report*, Los Alamos National Laboratory, 1996

GINGOLD R. A., MONAGHAN J. J., Smoothed particle hydrodynamics: theory and application to non-spherical stars, *Mon. Not. Roy. Astron. Soc.*, vol. 181, 1977, p. 375-389.

KUHNERT J., General smoothed particle hydrodynamics, Ph.D. thesis, Kaiserslautern University, Germany, 1999.

KUHNERT J., An upwind finite pointset method for compressible Euler and Navier-Stokes equations, In M. Griebel and M. A. Schweitzer, Meshfree Methods for Partial Differential Equations, Springer-Verlag.

KUHNERT J., TRAMECON A., ULLRICH P., Advanced Air Bag Fluid Structure Coupled Simulations applied to out-of Position Cases, *EUROPAM Conference Proceedings 2000*, ESI group, Paris, France

MONAGHAN J. J., Smoothed particle hydrodynamics, *Annu. Rev. Astron. Astrop*, vol. 30, 1992, p. 543-574.

MONAGHAN J. J., Simulating free surface flows with SPH, *J. Comput. Phys.*, vol. 110, 1994, p. 399.

MONAGHAN J. J., GINGOLD R. A., Shock Simulation by particle method SPH, *J. Comp. Phys.*, vol. 52, 1983, p. 374-389.

MORRIS J. P., FOX P. J., ZHU Y., Modeling Low Reynolds Number Incompressible Flows Using SPH, *J. Comput. Phys.*, vol. 136, 1997, p. 214-226.

TIWARI S., A LSQ-SPH approach for compressible viscous flows, *Proceedings of the 8th International Conference on Hyperbolic Problems Hyp2000*.

TIWARI S., MANSERVISI S., Modeling incompressible Navier-Stokes flows by LSQ-SPH, *Berichte des Fraunhofer ITWM*, Kaiserslautern, Germany, 2000.