MPM Extrapolations

September 29, 2016

1 The MPM Extrapolation Problem

Let lower case denote grid (or Eulerian frame) quantities and upper case denote particle (or Lagrangian frame) quantities. We define S to be any isoparametric grid-based shape function matrix to extrapolate from nodes to any position (normally to a particle position). Thus:

$$\begin{aligned} \boldsymbol{Q}_p &= \sum_i S_{pi}\boldsymbol{q}_i && (1)\\ 1 &= \sum_i S_{pi} && (2)\end{aligned}$$

The first extrapolates any nodal quantity, q_i, to a particle quantity, Q_p, and S_{pi} is shape function for node i at position of particle p (note that many MPM papers use S_{ip}, but the reversed order makes more sense with S and transformation from grid to particles). The second requires partition of unity for the shape functions. As a linear transformation

$$\boldsymbol{Q} = \mathsf{S}\boldsymbol{q} \tag{3}$$

where $\boldsymbol{Q}$ is vector of particle properties, $\boldsymbol{q}$ is vector of nodal quantities, and S is the $N \times n$ transformation matrix (N particles and n nodes).

The first task in MPM time step n is to extrapolate from the particles to the grid. We begin by assuming a linear transformation

$$\boldsymbol{v}_i^{(n)} = \sum_p S_{ip}^+ \boldsymbol{V}_p^{(n)} \qquad \text{or} \qquad \boldsymbol{v} = \mathsf{S}^+\boldsymbol{V} \tag{4}$$

where $\boldsymbol{v}$ and $\boldsymbol{V}$ are vectors of nodal and particle velocities, and S^+ is matrix of shape functions that functions as pseudo-inverse of S. To make these transformations invertible, we would require $\mathsf{S}\mathsf{S}^+ = \mathsf{I}_{p\times p}$, but in generally the matrices are not square and therefore cannot be inverted. We resort instead to a pseudo inverse. (Note" this relation will change when particle has spin.)

After extrapolating velocities to the grid MPM time step extrapolates particle stresses to the grid to find nodal forces, $\boldsymbol{f}_i^{(n)}$. The velocities and forces are used to update grid nodal positions and velocities using:

$$\begin{aligned} \tilde{\boldsymbol{x}}_i^{(n+1)} &= \boldsymbol{x}_i + \boldsymbol{v}_i^{(n)}\Delta t + \frac{1}{2}\boldsymbol{a}_i^{(n)}(\Delta t)^2 && (5)\\ \tilde{\boldsymbol{v}}_i^{(n+1)} &= \boldsymbol{v}_i^{(n)} + \boldsymbol{a}_i^{(n)}\Delta t && (6)\end{aligned}$$

The $\tilde{(\cdot)}$ nomenclature indicate grid position and velocity at the end of time step n, but it is not the grid position and velocity for time step $n+1$. Instead, for time step $n+1$, the grid position is reset to $\boldsymbol{x}_i$ and the velocity is extrapolated to the new grid using updated particle positions and velocities. The $\boldsymbol{a}_i^{(n)}$ is a grid acceleration given by $\boldsymbol{a}_i^{(n)} = \boldsymbol{f}_i^{(n)}/m_i^{(n)}$ where $m_i^{(n)}$ is nodal mass in time step n.

Next, the updated grid values are used to update particle positions and velocity, which can be written generally as

$$\begin{aligned} \boldsymbol{X}_p^{(n+1)} &= \boldsymbol{X}_p^{(n)} + \mathbb{V}_p^{(n)}\Delta t + \frac{1}{2}\mathbb{A}_p^{(n)}(\Delta t)^2 = \boldsymbol{X}_p^{(n)} + \mathbb{V}_p^{(n+1)}\Delta t - \frac{1}{2}\mathbb{A}_p^{(n)}(\Delta t)^2 && (7) \\ \boldsymbol{V}_p^{(n+1)} &= \boldsymbol{V}_p^{(n)} + \mathbb{A}_p^{(n)}\Delta t && (8) \end{aligned}$$

where $\mathbb{V}_p^{(n)}$ and $\mathbb{A}_p^{(n)}$ are velocity and acceleration extrapolated from the grid to the particle, by methods that are not yet determined or methods that can be potentially be selected to define various types of MPM (such as FLIP or PIC). Whatever justification is used to select them, however, they are subject the physical consistency restriction that:

$$\mathbb{V}_p^{(n+1)} = \mathbb{V}_p^{(n)} + \mathbb{A}_p^{(n)}\Delta t \tag{9}$$

1.1 Matrix Form

Try new form to fit in better's with Chad's approach. Let upper case vectors be particle properties and lower case grid properties. Let $\mathbb{V}^{(n)}$ and $\mathbb{A}^{(n)}$ be velocity and acceleration extrapolated to the particle (only these terms may change for various MPM methods such as FLIP, PIC, or XPIC) and assert that $\mathbb{V}^{(n)} = \mathsf{S}\boldsymbol{v}$. The task is to find $\mathbb{V}^{(n+1)}$ and $\mathbb{A}^{(n)}$ and use them for particle position and velocity updates. A generalized form for MPM is written as

$$\boldsymbol{V}^{(n+1)} = \mathsf{P}\boldsymbol{V}^{(n)} + \mathsf{S}\boldsymbol{a}\Delta t \tag{10}$$

where P is some projection tensor (and equal to I for FLIP, $\mathsf{S}\mathsf{S}^+$ for PIC, and $\beta\mathsf{I} + (1-\beta)\mathsf{S}\mathsf{S}^+$ for fraction β FLIP and $(1-\beta)$ PIC, and current MPM code uses $\mathsf{S}^+ = [m]^{-1}\mathsf{S}^T[M]$ where $[m]$ and $[M]$ are diagonal matrices with nodal and particle masses on the diagonals, respectively). By definition, the acceleration on the particle is

$$\mathbb{A}^{(n)}\Delta t = \boldsymbol{V}^{(n+1)} - \boldsymbol{V}^{(n)} = \mathsf{S}\boldsymbol{a}\Delta t - (\mathsf{I}-\mathsf{P})\boldsymbol{V}^{(n)} \tag{11}$$

Note that a second interpretation of P other then I leads to damping of the extrapolated acceleration or

$$\mathbb{A}^{(n)}\Delta t = \boldsymbol{V}^{(n+1)} - \boldsymbol{V}^{(n)} = \left(\mathsf{S}\boldsymbol{a} - \alpha(\mathsf{P})\boldsymbol{V}^{(n)}\right)\Delta t \tag{12}$$

where $\alpha(\mathsf{P}) = \frac{1}{\Delta t}(\mathsf{I}-\mathsf{P})$ is a damping term.

The effective updated velocity is

$$\mathbb{V}^{(n+1)} = \mathbb{V}^{(n)} + \mathbb{A}^{(n)}\Delta t = \mathsf{S}\boldsymbol{v} - (\mathsf{I}-\mathsf{P})\boldsymbol{V}^{(n)} + \mathsf{S}\boldsymbol{a}\Delta t = \mathsf{S}\tilde{\boldsymbol{v}} - (\mathsf{I}-\mathsf{P})\boldsymbol{V}^{(n)} \tag{13}$$

where $\tilde{\boldsymbol{v}} = \boldsymbol{v} + \boldsymbol{a}\Delta t$ is the updated nodal velocity. The final velocity and position updates become

$$\begin{aligned} \boldsymbol{V}^{(n+1)} &= \boldsymbol{V}^{(n)} + \mathbb{A}^{(n)}\Delta t = \mathsf{P}\boldsymbol{V}^{(n)} + \mathsf{S}\boldsymbol{a}\Delta t && (14) \\ \boldsymbol{X}^{(n+1)} &= \boldsymbol{X}^{(n)} + \mathbb{V}^{(n+1)}\Delta t - \frac{1}{2}\mathbb{A}^{(n)}(\Delta t)^2 && (15) \\ &= \boldsymbol{X}^{(n)} + \left[\mathsf{S}\tilde{\boldsymbol{v}} - (\mathsf{I}-\mathsf{P})\boldsymbol{V}^{(n)} - \frac{1}{2}\left(\mathsf{S}\boldsymbol{a}\Delta t - (\mathsf{I}-\mathsf{P})\boldsymbol{V}^{(n)}\right)\right]\Delta t && (16) \\ &= \boldsymbol{X}^{(n)} + \left[\mathsf{S}\left(\tilde{\boldsymbol{v}} - \frac{1}{2}\boldsymbol{a}\Delta t\right) - \frac{1}{2}(\mathsf{I}-\mathsf{P})\boldsymbol{V}^{(n)}\right]\Delta t && (17) \end{aligned}$$

Notice that code that uses these updates can accommodate any method that can be expressed with P.

The average Lagrangian velocity over the time step is

$$\frac{\boldsymbol{X}^{(n+1)}-\boldsymbol{X}^{(n)}}{\Delta t}=\mathsf{S}\left(\boldsymbol{v}+\frac{1}{2}\boldsymbol{a}\Delta t\right)-\frac{1}{2}(\mathsf{I}-\mathsf{P})\boldsymbol{V}^{(n)} \tag{18}$$

The average velocity over the time step is

$$\frac{\boldsymbol{V}^{(n)}+\boldsymbol{V}^{(n+1)}}{2}=\frac{1}{2}(\mathsf{I}+\mathsf{P})\boldsymbol{V}^{(n)}+\frac{1}{2}\mathsf{S}\boldsymbol{a}\Delta t=\frac{1}{2}(\mathsf{P}-\mathsf{I})\boldsymbol{V}^{(n)}+\boldsymbol{V}^{(n)}+\frac{1}{2}\mathsf{S}\boldsymbol{a}\Delta t \tag{19}$$

These should be equal in the Lagrangian view. The difference is an (unavoidable) MPM error:

$$(\text{error})=\boldsymbol{V}^{(n)}-\mathsf{S}\boldsymbol{v}=(\mathsf{I}-\mathsf{S}\mathsf{S}^{+})\boldsymbol{V}^{(n)} \tag{20}$$

This error suggest a path to improve FLIP is to damp this error by using

$$\mathbb{A}^{(n)}\ =\mathsf{S}\boldsymbol{a}-\alpha(\mathsf{I}-\mathsf{S}\mathsf{S}^{+})\boldsymbol{V}^{(n)} \tag{21}$$

Comparison to above leads to choice of

$$\mathsf{P}=(1-\alpha\Delta t)\mathsf{I}+\alpha\Delta t\mathsf{S}\mathsf{S}^{+} \tag{22}$$

which is recognized as a re-invention of hybrid FLIP/PIC simulation using fraction $\alpha\Delta t$ of PIC. Three differences from the usual FLIP/PIC description are that:

1. PIC is not a new method, but rather MPM is FLIP and PIC is a damped version of FLIP.
2. This P leads to a required position update of:

$$\boldsymbol{X}^{(n+1)}=\boldsymbol{X}^{(n)}+\left[\mathsf{S}\left(\tilde{\boldsymbol{v}}-\frac{1}{2}\boldsymbol{a}\Delta t\right)-\frac{\alpha}{2}(\mathsf{I}-\mathsf{S}\mathsf{S}^{+})\boldsymbol{V}^{(n)}\right]\Delta t \tag{23}$$

3. A common error by those using mixed PIC and FLIP is to think a first order position update version for mixed FLIP and PIC is

$$\boldsymbol{X}^{(n+1)}=\boldsymbol{X}^{(n)}+\mathsf{S}\tilde{\boldsymbol{v}}\Delta t \tag{24}$$

   Compounding this error is the misleading observation that this scheme provides an "exact" Lagrangian update such that $d\boldsymbol{X}^{(n+1)}/dt=\boldsymbol{V}^{(n+1)}$. but, be careful of what you get with "exact" results, because this first order does not have the entire first order term. You either need both first order terms or the full second order update. Although this is no longer an exact Lagrangian update, it is the required update when using any PIC character. See my cutting paper for why it is needed.

2 Current MPM Methods

2.1 Extrapolation to the Grid

Sulsky derived the MPM method using a weighted least square assumption. Assuming particle velocities extrapolated from grid velocities is given by $\boldsymbol{V}^{(n+1)}_{g\to p}=\sum_i S_{pi}\boldsymbol{v}^{(n)}_i$, find the $\boldsymbol{v}^{(n)}_i$ to minimize the difference of this velocity with initial particle velocity ($\boldsymbol{V}_p^{(n)}$) weighted by the particle mass:

$$\begin{aligned}
\Omega &= \sum_p M_p\big(\boldsymbol{V}_p^{(n)} - \boldsymbol{V}_{g\to p}^{(n)}\big)^2 = \sum_p M_p\big(\boldsymbol{V}_p^{(n)} - \sum_i S_{pi}\boldsymbol{v}_i^{(n)}\big)^2 && (25)\\
0 = \frac{d\Omega}{d\boldsymbol{v}_j^{(n)}} &= -2\left(\sum_p S_{pj}M_p\boldsymbol{V}_p^{(n)} - \sum_i\sum_p M_pS_{pj}S_{pi}\boldsymbol{v}_i^{(n)}\right) && (26)\\
\sum_i m_{ji}^{(n)}\boldsymbol{v}_i^{(n)} &= \sum_p S_{pj}P_p^{(n)} && (27)\\
\mathsf{m}\boldsymbol{v} &= \mathsf{S}^T\boldsymbol{P} && (28)\\
\boldsymbol{v} &= \mathsf{m}^{-1}\mathsf{S}^T\boldsymbol{P} && (29)
\end{aligned}$$

where $m_{ji}^{(n)} = \sum_p M_pS_{pj}S_{pi}$ is an element of the full mass matrix m and $\boldsymbol{P}$ is a vector of particle momenta. If we replace m by the diagonal lumped mass matrix with $\boldsymbol{m}_i^{(n)} = \sum_j m_{ij}^{(n)} = \sum_p M_pS_{pi}$, this result reduces to

$$S_{ip}^{+} = \frac{M_pS_{pi}}{\boldsymbol{m}_i^{(n)}} \tag{30}$$

Note that the GIMP derivation by Bardenhagen and Kober gets the same result without the needing to assert a lumped mass matrix (this fact may or may not be significant).

2.2 FLIP Particle Updates

The undamped FLIP method is recovered by setting $\mathsf{P} = \mathsf{I}$ to give

$$\mathbb{V}^{(n)} = \mathsf{S}\boldsymbol{v}, \qquad \mathbb{V}^{(n+1)} = \mathsf{S}\tilde{\boldsymbol{v}}, \qquad \text{and} \qquad \mathbb{A}^{(n)} = \mathsf{S}\boldsymbol{a} \tag{31}$$

The particle updates become

$$\begin{aligned}
\boldsymbol{V}^{(n+1)} &= \boldsymbol{V}^{(n)} + \mathsf{S}\boldsymbol{a}\Delta t && (32)\\
\boldsymbol{X}^{(n+1)} &= \boldsymbol{X}^{(n)} + \left[\mathsf{S}\left(\tilde{\boldsymbol{v}} - \frac{1}{2}\boldsymbol{a}\Delta t\right)\right]\Delta t && (33)
\end{aligned}$$

Note that equating accelerations

$$\boldsymbol{V}^{(n+1)} - \mathsf{S}\tilde{\boldsymbol{v}} = \boldsymbol{V}^{(n)} - \mathsf{S}\boldsymbol{v} \tag{34}$$

Writing out the summations gives components of $\mathbb{V}^{(n)}$, $\mathbb{V}^{(n+1)}$, $\mathbb{A}^{(n)}$, $\boldsymbol{X}$, and $\boldsymbol{V}$:

$$\begin{aligned}
\mathbb{V}_p^{(n)} &= \boldsymbol{V}_{g\to p}^{(n)} = \sum_i S_{pi}\boldsymbol{v}_i^{(n)} && (35)\\
\mathbb{A}_p^{(n)} &= \boldsymbol{A}_{g\to p}^{(n)} = \sum_i S_{pi}\boldsymbol{a}_i^{(n)} && (36)\\
\mathbb{V}_p^{(n+1)} &= \boldsymbol{V}_{g\to p}^{(n+1)} = \sum_i S_{pi}\tilde{\boldsymbol{v}}_i^{(n+1)} = \boldsymbol{V}_{g\to p}^{(n)} + \boldsymbol{A}_{g\to p}^{(n)}\Delta t && (37)\\
\boldsymbol{X}_p^{(n+1)} &= \boldsymbol{X}_p^{(n)} + \boldsymbol{V}_{g\to p}^{(n+1)}\Delta t - \frac{1}{2}\boldsymbol{A}_{g\to p}^{(n)}(\Delta t)^2 && (38)\\
\boldsymbol{V}_p^{(n+1)} &= \boldsymbol{V}_p^{(n)} + \boldsymbol{A}_{g\to p}^{(n)}\Delta t && (39)
\end{aligned}$$

and note that

$$\boldsymbol{V}_{g\to p}^{(n+1)} - \boldsymbol{V}_p^{(n+1)} = \boldsymbol{V}_{g\to p}^{(n)} - \boldsymbol{V}_p^{(n)} \tag{40}$$

2.3 PIC Particle Updates

The undamped PIC method is recovered by setting $\mathsf{P} = \mathsf{S}\mathsf{S}^{+}$ to give

$$\mathbb{V}^{(n)} = \mathsf{S}\boldsymbol{v}, \qquad \mathbb{V}^{(n+1)} = \mathsf{S}\tilde{\boldsymbol{v}} - (\mathsf{I} - \mathsf{S}\mathsf{S}^{+})\boldsymbol{V}^{(n)} = \mathsf{S}\left(\tilde{\boldsymbol{v}} + \boldsymbol{v}\right) - \boldsymbol{V}^{(n)} \tag{41}$$

and acceleration reduces to

$$\mathbb{A}^{(n)}\Delta t = \mathsf{S}\boldsymbol{a}\Delta t - (\mathsf{I} - \mathsf{S}\mathsf{S}^{+})\boldsymbol{V}^{(n)} = \mathsf{S}\tilde{\boldsymbol{v}} - \boldsymbol{V}^{(n)} \tag{42}$$

The updates become

$$\begin{aligned}
\boldsymbol{V}^{(n+1)} &= \mathsf{S}\mathsf{S}^{+}\boldsymbol{V}^{(n)} + \mathsf{S}\boldsymbol{a}\Delta t = \mathsf{S}\tilde{\boldsymbol{v}} && (43)\\
\boldsymbol{X}^{(n+1)} &= \boldsymbol{X}^{(n)} + \left[\mathsf{S}\left(\tilde{\boldsymbol{v}} - \frac{1}{2}\boldsymbol{a}\Delta t\right) - \frac{1}{2}(\mathsf{I} - \mathsf{S}\mathsf{S}^{+})\boldsymbol{V}^{(n)}\right]\Delta t && (44)\\
&= \boldsymbol{X}^{(n)} + \left[\mathsf{S}\left(\tilde{\boldsymbol{v}} + \frac{1}{2}\boldsymbol{v} - \frac{1}{2}\boldsymbol{a}\Delta t\right) - \frac{1}{2}\boldsymbol{V}^{(n)}\right]\Delta t && (45)\\
&= \boldsymbol{X}^{(n)} + \left[\mathsf{S}\boldsymbol{v} + \frac{\mathsf{S}\tilde{\boldsymbol{v}} - \boldsymbol{V}^{(n)}}{2}\right]\Delta t && (46)
\end{aligned}$$

This approach derives and unexpected position update. Although first order in time, it could not be derived without using second order update in the generalized method.

Writing out the summations gives the components:

$$\begin{aligned}
\mathbb{V}_p^{(n)} &= \boldsymbol{V}_{g\to p}^{(n)} = \sum_i S_{pi}\boldsymbol{v}_i^{(n)} && (47)\\
\mathbb{A}_p^{(n)}\Delta t &= \boldsymbol{V}_{g\to p}^{(n+1)} - \boldsymbol{V}_p^{(n)} && (48)\\
\mathbb{V}_p^{(n+1)} = \mathbb{V}_p^{(n)} + \mathbb{A}_p^{(n)}\Delta t &= \boldsymbol{V}_{g\to p}^{(n+1)} + \boldsymbol{V}_{g\to p}^{(n)} - \boldsymbol{V}_p^{(n)} && (49)\\
\boldsymbol{V}_p^{(n+1)} &= \boldsymbol{V}_{g\to p}^{(n+1)} && (50)\\
\boldsymbol{X}_p^{(n+1)} &= \boldsymbol{X}_p^{(n)} + \left(\boldsymbol{V}_{g\to p}^{(n)} + \frac{\boldsymbol{V}_p^{(n+1)} - \boldsymbol{V}_p^{(n)}}{2}\right)\Delta t && (51)
\end{aligned}$$

2.4 FLIP/PIC Particle Updates Generalized

A combined FLIP and PIC update can be expressed as a damped FLIP analysis using

$$\begin{aligned}
\mathbb{V}_p^{(n)} &= \boldsymbol{V}_{g\to p}^{(n)} = \sum_i S_{pi}\boldsymbol{v}_i^{(n)} && (52)\\
\mathbb{A}_p^{(n)} &= \sum_i S_{pi}\boldsymbol{a}_i^{(n)} - \frac{1-\beta}{\Delta t}\left(\boldsymbol{V}_p^{(n)} - \boldsymbol{V}_{g\to p}^{(n)}\right) && (53)\\
\mathbb{V}_p^{(n+1)} = \mathbb{V}_p^{(n)} + \mathbb{A}_p^{(n)}\Delta t &= \boldsymbol{V}_{g\to p}^{(n+1)} - (1-\beta)\left(\boldsymbol{V}_p^{(n)} - \boldsymbol{V}_{g\to p}^{(n)}\right) && (54)
\end{aligned}$$

leading to

$$\begin{aligned}
\boldsymbol{V}_p^{(n+1)} &= \boldsymbol{V}_p^{(n)} - (1-\beta)(\boldsymbol{V}_p^{(n)} - \boldsymbol{V}_{g\to p}^{(n)}) + \boldsymbol{A}_{g\to p}^{(n)}\Delta t && (55)\\
\boldsymbol{X}_p^{(n+1)} &= \boldsymbol{X}_p^{(n)} + \left(\boldsymbol{V}_{g\to p}^{(n)} - \frac{(1-\beta)}{2}(\boldsymbol{V}_p^{(n)} - \boldsymbol{V}_{g\to p}^{(n)})\right)\Delta t + \frac{1}{2}\boldsymbol{A}_{g\to p}^{(n)}(\Delta)^2 && (56)\\
&= \boldsymbol{X}_p^{(n)} + \frac{1}{2}\left(\boldsymbol{V}_{g\to p}^{(n+1)} + \boldsymbol{V}_{g\to p}^{(n)} - (1-\beta)(\boldsymbol{V}_p^{(n)} - \boldsymbol{V}_{g\to p}^{(n)})\right)\Delta t && (57)
\end{aligned}$$

2.5 Combined FLIP/PIC Implementation with Damping

The particle acceleration for FLIP/PIC with optional damping and linear addition of FLIP

$$\begin{aligned} \mathbb{A}^{(n)}\Delta t &= \mathsf{S}\boldsymbol{a}\Delta t - (1-\beta)(\mathsf{I}-\mathsf{S}\mathsf{S}^{+})\boldsymbol{V}^{(n)} - \alpha_g \Delta t \mathsf{S}\boldsymbol{v} - \alpha_p \Delta t \boldsymbol{V}^{(n)} && (58) \\ &= \mathsf{S}\boldsymbol{a}\Delta t - \big(\alpha_p \Delta t + (1-\beta)\big)\boldsymbol{V}^{(n)} - \big(\alpha_g \Delta t - (1-\beta)\big)\mathsf{S}\boldsymbol{v} && (59) \end{aligned}$$

When the particle update is done, the nodes will have only updated velocity. Using $\tilde{\boldsymbol{v}} = \boldsymbol{v} + \boldsymbol{a}\Delta t$, the acceleration becomes

$$\begin{aligned} \mathbb{A}^{(n)} &= \big(1+\alpha_g^{(tot)}\Delta t\big)\mathsf{S}\boldsymbol{a} - \alpha_p^{(tot)}\boldsymbol{V}^{(n)} - \alpha_g^{(tot)}\mathsf{S}\tilde{\boldsymbol{v}} && (60) \\ \alpha_g^{(tot)} &= \alpha_g - \frac{m(1-\beta)}{\Delta t} \quad \text{and} \quad \alpha_p^{(tot)} = \alpha_p + \frac{(1-\beta)}{\Delta t} && (61) \end{aligned}$$

The updates become

$$\begin{aligned} \boldsymbol{V}^{(n+1)} &= (1-\alpha_p^{(tot)}\Delta t)\boldsymbol{V}^{(n)} + \Big(\big(1+\alpha_g^{(tot)}\Delta t\big)\mathsf{S}\boldsymbol{a} - \alpha_g^{(tot)}\mathsf{S}\tilde{\boldsymbol{v}}\Big)\Delta t && (62) \\ \boldsymbol{X}^{(n+1)} &= \boldsymbol{X}^{(n)} + \mathsf{S}\tilde{\boldsymbol{v}}\Delta t - \Big(\big(1-\alpha_g^{(tot)}\Delta t\big)\mathsf{S}\boldsymbol{a} + \alpha_g^{(tot)}\mathsf{S}\tilde{\boldsymbol{v}} + \alpha_p^{(tot)}\boldsymbol{V}^{(n)}\Big)\frac{(\Delta t)^2}{2} && (63) \end{aligned}$$