This is a repository copy of *Using the polynomial particle-in-cell method for liquid-fabric interaction*.

White Rose Research Online URL for this paper:
https://eprints.whiterose.ac.uk/209828/

Version: Published Version

Using the Polynomial Particle-in-Cell Method for Liquid-Fabric Interaction

Robert Dennison and Steve Maddock

The University of Sheffield, U.K.

Keywords: Physically-Based Modeling, Fluid Simulation, Cloth Simulation.

Abstract: Liquid-fabric interaction simulations using particle-in-cell (PIC) based models have been used to simulate a wide variety of phenomena and yield impressive visual results. However, these models suffer from numerical damping due to the data interpolation between the particles and grid. Our paper addresses this by using the polynomial PIC (PolyPIC) model instead of the affine PIC (APIC) model that is used in current state-of-the-art wet cloth models. Theoretically, PolyPIC has lossless energy transfer and so should avoid any problems of undesirable damping and numerical viscosity. Our results show that PolyPIC does enable more dynamic coupled simulations. The use of PolyPIC allows for simulations with reduced numerical dissipation and improved resolution of vorticial details over previous work. For smaller scale simulations, there is minimal impact on computational performance when using PolyPIC instead of APIC. However, as simulations involve a larger number of particles and mesh elements, PolyPIC can require up to a $2.5\times$ as long to generate $4.0s$ of simulation due to a requirement for a decrease in timestep size to remain stable.

1 INTRODUCTION

In physics-based simulations for computer graphics, increasing attention is being focused on simulations involving a combination of two or more physical media. One area of particular interest is simulating the complex interactions between fluid and porous fabric. The liquid-fabric interaction model of Fei et al (Fei et al., 2018) is widely considered as the state-of-the-art for this kind of coupled simulation. They make use of the affine particle-in-cell (APIC) model for the fluids and the material point method (MPM) using APIC transfers for cloth and yarn objects.

Particle-in-cell (PIC) methods are a class of hybrid Eulerian-Lagrangian simulation methods, designed to benefit from the ease of particle advection found in Lagrangian methods with the simplicity of calculating forces and material properties on a regular Eulerian grid. In each time step, particle information is interpolated to nearby grid nodes where an updated velocity can be calculated. This new velocity can then be interpolated back to the particles which can be advected through the simulation domain. A known issue with PIC methods is that they suffer from numerical dissipation, as the interpolation stages act as a filter of high frequency and rotational velocities. This means that fluid simulations using PIC methods can seem

overly viscous. APIC was introduced as an improvement to the standard PIC model (Jiang et al., 2015) and was developed to reduce the numerical dissipation by considering rotational velocity. Since its introduction, APIC has seen wide adoption by the visual effects industry.

While APIC was developed in an effort to reduce damping of rotational velocities, it has been shown to still be introduce numerical damping. This makes it more challenging to use real world values for viscosity and elasticity, as the model will introduce viscosity numerically. The polynomial particle-in-cell model (PolyPIC) was introduced as a generalised extension of APIC to further reduce numerical dissipation (Fu et al., 2017), and has been shown to theoretically be able to achieve lossless energy transfers during the interpolation stages. By using a method with reduced energy loss, real world empirical data can be used as the model parameters, allowing for easier recreation of real world scenes by artists.

Our paper is the first to use PolyPIC for liquid-fabric interactions. We improve the work of Fei et al by incorporating polynomial transfers between the particles and grid. By substituting the affine transfers of APIC for higher order polynomials, the numerical damping of the model can be reduced. This also requires an alteration of some aspects of the pre-
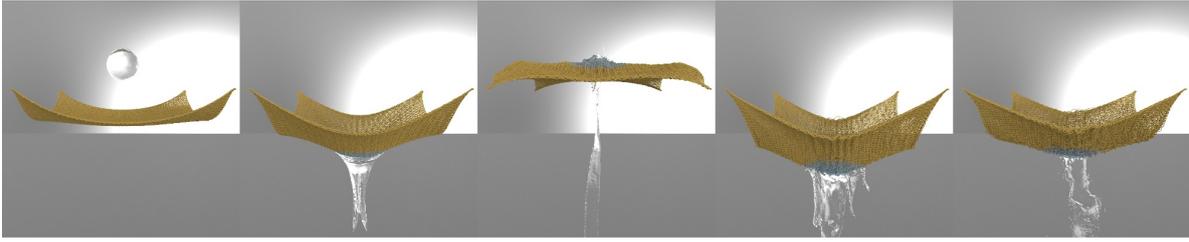
Figure 1: Fluid splashing onto a square of yarn fabric using PolyPIC transfers.

sented PolyPIC method to improve simulation stability. The results give a comparison of PolyPIC with APIC for coupled simulation scenarios and demonstrate that PolyPIC provides improved preservation of rotational velocity, leading to more dynamic simulations. The presented scenarios give a comparison of APIC and PolyPIC in a pure fluid simulation to showcase the differences of the two simulation approaches.

The remainder of this paper is structured into five sections. Section 2 describes related work. Section 3 explains the implementation of PolyPIC for the liquid-fabric simulation. The results in section 4 provide a comparison of PolyPIC and APIC, with a discussion in section 5. Section 6 presents conclusions.

2 RELATED WORK

Early work on wet cloth models focused on non-porous thin shells interacting with fluids (e.g. (Harada et al., 2007)), with subsequent work involving porous volumes (Lenaerts et al., 2008). The main drawback of this work was that it used a smoothed particle hydrodynamics framework, thus requiring a large number of particles internal to the object's volume and impacting on computational performance. Huber et al (Huber et al., 2011) improved the computational performance by using a cellular automata approach which allowed for simple parallelization. Patkar et al (Patkar and Chaudhuri, 2013) used a mesh-based approach for solid simulation to remove the need for simulating large numbers of solid particles and also incorporated additional fluid effects such as dripping and surface flow.

Further developments came with the application of position based dynamics (Müller et al., 2007) to the model presented by Lenaerts et al, thus allowing for larger time steps to be used (Shao et al., 2018). Another recent development by Fei et al (Fei et al., 2018) demonstrates both a porous cloth model using discrete shells (Grinspun et al., 2003) and discrete rods (Bergou et al., 2008) discretized using the material point method (MPM) (Sulsky et al., 1994) for cloth and yarn simulations, respectively. Zheng et al

then added further realism to liquid-yarn interactions by modelling the defects in the constituent fibres of a material (Zheng et al., 2021). However, this work has currently only been used to model static fabric.

Particle-in-cell methods have seen extensive use since first being introduced by Harlow et al (Harlow, 1962). The severe numerical dissipation of this method due to filtering caused by the number of interpolation steps led to the development of the fluid implicit particle (FLIP) method, which bypassed some of the interpolation steps to increase the dynamism of simulations by interpolating only the change in velocity from the grid to the particles rather than interpolating the velocity (Brackbill and Ruppel, 1986). The major drawback of the FLIP method was that reduced dissipation came at the cost of introducing more noise and instability to the simulations.

Jiang et al (Jiang et al., 2015) developed the APIC method as a way of reducing the numerical dissipation of PIC in a stable way by storing an affine transform matrix on each particle, as well as a velocity vector. This reduced the energy loss of the PIC method and improved the stability of FLIP, and also improved the preservation of angular momentum of both previous iterations of the model. Recently PolyPIC (Fu et al., 2017) was developed as a generalized extension of APIC to allow for higher order transfers during the grid/particle transfers, further reducing the numerical dissipation and improving preservation of angular momentum. It achieves this by replacing the affine matrix used to store information about angular momentum with a more general polynomial function.

The popularity of PIC methods led to their application to deformable materials by Sulsky et al to create MPM (Sulsky et al., 1994). Subsequent PIC developments have also continued to be adapted to elastic solids to continually improve MPM simulations, and MPM has been successfully applied to simulate a wide variety of materials such as snow (Stomakhin et al., 2013), viscoelastic solids (Fang et al., 2019) and even materials with phase changes such as lava or butter (Stomakhin et al., 2014). Despite the improvements of PolyPIC over APIC, PolyPIC has not yet been adapted to coupled simulation scenarios.

3 METHOD

This paper builds on the work of Fei et al by replacing the APIC model with the PolyPIC model. APIC and PolyPIC differ only in the transfer steps (as shown in Figure 2), meaning PolyPIC transfers can be substituted into the model in the place of APIC transfers. Section 3.1 presents the standard PIC model, section 3.2 describes how APIC builds upon this and PolyPIC is explained in section 3.3. Finally, section 3.4 describes the application of PolyPIC to a mixture model. Fei et al used a staggered marker-and-cell (MAC) grid approach as it provides more accurate central differences over a standard collocated grid approach, so all work presented here is applied to MAC grids.
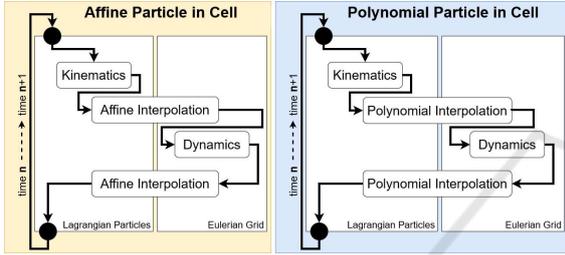

Figure 2: The difference between the APIC and PolyPIC algorithms is the method of transferring data between the particles and the grid. PolyPIC uses generalized higher order polynomials whereas APIC uses affine transformation matrices. (Based on Figure 7 in (Jiang et al., 2015)).

3.1 The Particle-in-Cell Method

The standard PIC model consists of particles which store information about their mass and velocity, which is interpolated to/from the grid node faces to advect the particles around the simulation domain, as shown in Figure 3. The yellow and blue boxes will be considered in subsections 3.2 and 3.3, respectively.
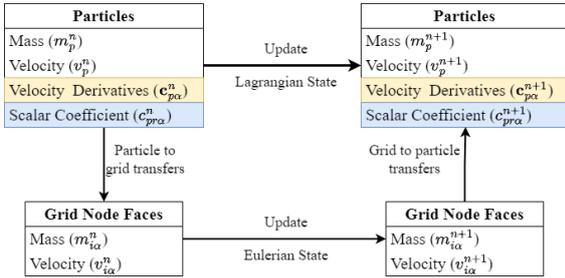

Figure 3: The data stored by particles and grid node faces in PIC methods. White background boxes are consistent between all PIC variants. Velocity derivatives (yellow background) are introduced for APIC and scalar coefficients (blue boxes) are introduced in PolyPIC. (Based on a diagram in (Fu et al., 2017)).

Table 1: Notation used throughout this paper. Values are of type scalar s, vector v or matrix m.

Notation	Type	Meaning
Δt	s	size of time step
Δx	s	size of grid node
x_p	v	position of particle p
$x_{i\alpha}$	v	position of face α of grid node i
m_p	s	mass of particle p
v_p	v	velocity of particle p
$m_{i\alpha}$	s	mass of face α of grid node i
$v_{i\alpha}$	v	velocity of face α of grid node i
$\tilde{v_{i\alpha}}$	v	intermediate velocity of face α of grid node i
$w_{ip\alpha}$	s	contribution of particle p to face α of grid node i
$m_{ip\alpha}$	s	mass contributed by particle p to face α of grid node i
$(mv)_{ip\alpha}$	v	momentum contributed by particle p to face α of grid node i
$\mathbf{C}_p$	m	velocity derivatives of particle p, as described in (Jiang et al., 2015)
$\mathbf{c}_{p\alpha}$	v	velocity derivatives used for MAC grids
$c_{pr\alpha}$	s	scalar coefficient of scalar mode r for particle p in axis α
$\mathbf{I}$	m	identity matrix
$\mathbf{e}_\alpha$	v	basis vector for axis α

A PIC simulation consists of a total of P particles and N_i grid nodes, where superscript n is used for a given quantity at the current timestep n (e.g. v_p^n is the velocity of particle p at timestep n) and α represents a face of a grid node for each axis $1 \leq \alpha \leq d$, where d is the number of dimensions. The notation used throughout this paper is summarised in Table 1.

In the standard PIC method, each particle stores its own mass and velocity. A weight function $N(x)$ is defined to calculate the contribution of a particle to nearby grid cells during the interpolation steps and vice versa. Each particle within a nearby region of a grid node contributes some of its mass to each face (Equation 1). The mass contribution of each particle can then be multiplied by the particle's velocity to calculate the momentum contribution of each particle (Equation 2). The momentum contribution can then be summed over all particles to calculate the total momentum for the grid node face (Equation 3). The total mass of the grid node face is the mass contributions of all particles (Equation 4). Finally, the velocity of each grid node face can be calculated by dividing the momentum by the mass (Equation 5).

$$m_{ip\alpha}^n = m_p^n w_{ip\alpha} \tag{1}$$

$$(mv)_{ip\alpha}^n = m_{ip\alpha}^n v_p \tag{2}$$

$$(mv)_{i\alpha}^n = \sum_{p=0}^{P} (mv)_{ip\alpha}^n \tag{3}$$

$$m_{i\alpha}^n = \sum_{p=0}^{P} m_{ip\alpha}^n \tag{4}$$

$$v_{i\alpha}^n = (mv)_{i\alpha}^n / m_{i\alpha}^n \tag{5}$$

where $w_{ip\alpha} = N(x_{p\alpha} - x_{i\alpha})$ is the contribution of particle p to face α of grid node i.

The mass of a grid node can change throughout the simulation, but the masses of particles are fixed. This means when interpolating from the grid nodes to the particles, we only update the particles' velocities. Once the intermediate grid velocities $\tilde{v_{i\alpha}}^{n+1}$ have been calculated, they are interpolated back to the particles. To calculate the contribution of grid node momentum to each particle, first the momentum of each grid node must be calculated. This is done by multiplying the mass of each node face by the intermediate velocity of each node face and summing over the number of faces (Equation 6). Then the momentum of each particle can be calculated by summing the contributed momentum of each grid node over the number of grid nodes (Equation 7). Finally, the velocity of each particle can be calculated by dividing the momentum by the particle mass (Equation 8).

$$(mv)_{ip}^{n+1} = \sum_{\alpha=1}^{d} m_{ip\alpha}^{n} \tilde{v_{i\alpha}}^{n+1} \tag{6}$$

$$(mv)_{p}^{n+1} = \sum_{i=0}^{N_i} (mv)_{ip}^{n+1} \tag{7}$$

$$v_{p}^{n+1} = (mv)_{p}^{n+1}/m_p \tag{8}$$

3.2 The Affine PIC Method

In the APIC method, alongside mass and velocity each particle also stores an affine matrix of velocity derivatives to enhance the preservation of rotational velocities (see Figure 3). Equations 1 and 4, used for transferring mass from the particles to the grid nodes, remain unchanged for APIC. The updated momentum transfer takes into account the velocity derivatives to preserve angular momentum (Equation 9 (Equation 13 in Jiang et al)). The grid momentum can calculated as with PIC using Equation 3.

$$(mv)_{ip\alpha}^{n} = m_{ip\alpha}(\mathbf{e}_\alpha v_p + \mathbf{c}_{p\alpha}^{T}(x_p - x_{i\alpha})) \tag{9}$$

where $m_{ip\alpha}$ is calculated using Equation 1.

The momentum transfers from the grid nodes to the particles are the same as those used in the standard PIC method given by Equations 6 - 8. At this stage the velocity derivatives of a particle, $\mathbf{c}_{p\alpha}$, are updated by calculating the gradient of the weights relating that particle to the grid node faces, $\nabla w_{ip\alpha}$, multiplied by the intermediate velocity, then summing over all grid nodes (Equation 10 (Equation 14 in Jiang et al)).

$$\mathbf{c}_{p\alpha}^{n+1} = \sum_{i}^{N_i} \nabla w_{ip\alpha} \tilde{v_{i\alpha}}^{n+1} \tag{10}$$

3.3 The Polynomial PIC Method

PolyPIC replaces the affine matrix of APIC with generalized polynomials which allow for a wider range of local behaviour capture (see Figure 3). For PolyPIC the number of modes used is given by N_r. It uses polynomials of the form:

$$s(\mathbf{z}) = \prod_{\beta=1}^{d} z_\beta^{i_\beta} \tag{11}$$

where z_β is the βth component of $\mathbf{z} \in \mathbb{R}^d$ with $i_\beta \in \mathbb{Z}^+$.

Before defining the transfers, we must first define a map of the simulation configuration at time t^{n+1} to t^n, denoted as $\xi^{n+1}(x)$. As described by Fu et al, this map can take different forms for constant or affine material motion. Here, the affine map is given by Equation 12 (Equation 10 in Fu et al).

$$\xi^{n+1}(\mathbf{x}) = \mathbf{x}_p^n + (I + \Delta t C_p^{n+1})^{-1}(\mathbf{x} - \mathbf{x}_p^{n+1}) \tag{12}$$

The method of calculating $\mathbf{C}_p^n$ is given in detail in (Jiang et al., 2015). The momentum transfer from the particles to the grid is then calculated by taking the sum of all the scalar modes multiplied by the corresponding scalar coefficients, given by Equation 13 (Equation 11 in Fu et al). The momentum contribution of each particle to the grid node faces can then be summed over all particles (Equation 3). The velocity of each grid node face can then be calculated by dividing the momentum by the mass (Equation 5).

$$(mv)_{ip\alpha}^{n} = m_{ip\alpha}^{n} \sum_{r=0}^{N_r} s_r(\xi_p^n(x_{i\alpha} - x_p^{n-1})) c_{pr\alpha}^{n} \tag{13}$$

where s_r is scalar mode r. The coefficients $c_{pr\alpha}^{n}$ are calculated as a minimisation problem as described by Fu et al. To efficiently calculate the coefficients, the resultant linear system requires each dimension to be decoupled. The coefficients for modes $1 \leq r \leq 2^d$ are naturally mass-orthogonal and so solutions can be efficiently found. However, higher order coefficients $2^d < r \leq N_r$ require modification to be orthogonalized. This is achieved by substituting the quadratic terms, z_β, in Equation 11 with $g_\beta(z_\beta)$ defined in Equation 14. In contrast to the equation presented in Fu et al, the variables in Equation 14 are modified to orthogonalize the matrix, which produces more stable simulations.

$$g_\beta(z_\beta) = (w_{ip}^n)^2 - w_{ip}^n \frac{z_\beta(\Delta x^2 - 4z_\beta^2)}{\Delta x^2} - \frac{\Delta x^2}{4} \tag{14}$$

Again, the momentum transfers from the grid to the particles are the same as those used in the standard PIC method given by Equations 6, 7 and 8. At

this stage, the velocity derivatives C_p can be calculated using the method described by Jiang et al and the coefficients $c_{pr\alpha}$ can be calculated as described by Fu et al.

3.4 Liquid-Fabric Interaction

The model presented by Fei et al relies on the use of mixture theory to simulate the interactions between fluid and cloth (Nielsen and Østerby, 2013). Therefore, the momentum transfers need to be altered in order to be applied to mixtures. Let $m_{s,p}^n$ be the mass of a solid particle p at time n and $m_{f,p}^n$ be the mass of a fluid particle p at time n. The fluid transfers are then the same as those given in Equations 3, 5 and 13, except only particles tagged as fluids are considered (Equations 15 - 17).

$$(m_f v_f)_{ip\alpha}^n = m_{f,ip\alpha}^n \sum_{r=0}^{N_r} s_r (\xi_p^n (x_{i\alpha} - x_p^{n-1}) c_{pr\alpha}^n \quad (15)$$

$$(m_f v_f)_{i\alpha}^n = \sum_p (m_f v_f)_{ip\alpha}^n \quad (16)$$

$$v_{f,i\alpha}^n = (m_f v_f)_{i\alpha}^n / m_{f,i\alpha}^n \quad (17)$$

The solid transfers are very similar to the fluid transfers, except that absorbed fluid particles must be considered. To take account of absorption, we must also consider the mass of absorbed fluid particles when transferring the momentum of solids to the grid. The total mass of a solid particle is therefore given by summing the mass of the solid particle itself and the mass of the absorbed fluid particle, $(m_{s,ip\alpha}^n + m_{f,ip\alpha}^n)$. These solid-fluid mixture momentum transfers are given in Equations 18 - 20.

$$(m_s v_s)_{ip\alpha}^n = (m_{s,ip\alpha}^n + m_{f,ip\alpha}^n) \sum_{r=0}^{N_r} s_r (\xi_p^n (x_{i\alpha} - x_p^{n-1}) c_{pr\alpha}^n \quad (18)$$

$$(m_s v_s)_{i\alpha}^n = \sum_p (m_s v_s)_{ip\alpha}^n \quad (19)$$

$$v_{s,i\alpha}^n = (m_s v_s)_{i\alpha}^n / (m_{s,ip\alpha}^n + m_{f,ip\alpha}^n) \quad (20)$$

4 RESULTS

Four scenarios have been used to test the model that has been developed:

- Figure 4 shows a simple dam break scenario to highlight the differences between APIC fluids and PolyPIC fluids.
- Figure 5 shows a ball of fluid falling onto a small square of cloth draped over a sphere.

- Figure 6 shows a ball of fluid falling onto a small square of yarn-based fabric draped over a sphere.
- Figure 7 is similar to the small yarn scenario but shows a larger volume of liquid falling onto a medium sized square of yarn-based fabric.

The latter three scenarios were used by Fei et al, enabling a comparison of the original model using APIC and the new model using PolyPIC. In the simulation results, particle colours are based on current particle velocity, where dark blue represents a low velocity and red a high velocity. Videos of each simulation can be found in the supplementary video.

The simple dam break scenario (Figure 4) highlights the difference between PolyPIC and APIC outside of a coupled simulation context. This shows that using PolyPIC increases the conservation of angular momentum and so improves the resolution of vorticial detail of the fluid. As the final frame shows, PolyPIC particles have higher velocities than in APIC showing that numerical damping is reduced when using PolyPIC.
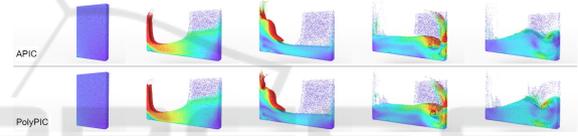

Figure 4: A small dam break scenario using APIC (top) and PolyPIC (bottom) transfers. Particle colours indicate velocity (dark blue = low, red = high). PolyPIC shows improved resolution of vorticial details.

A ball of fluid falling onto a small square of cloth fabric is shown in Figure 5. This example demonstrates the reduced numerical damping of PolyPIC causes less of the fluid to be absorbed by the cloth, as more splashes off as it is falling. The fluid that is absorbed exhibits more thin strand behaviour as it drips from the cloth in PolyPIC than APIC.
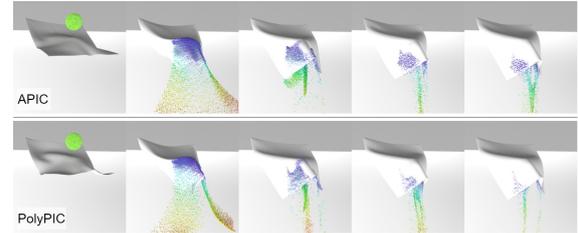

Figure 5: Fluid splash onto a small piece of cloth using APIC (top) and PolyPIC (bottom) transfers. Particle colours indicate velocity (dark blue = low, red = high). The reduced damping means that more fluid splashes off the piece of cloth rather than being absorbed. The fluid that is absorbed exhibits more thin strand behaviour as it drips from the cloth in PolyPIC than APIC (see frame 4).

A ball of fluid falling onto a small square of yarn fabric is shown in Figure 6. This scenario shows that the PolyPIC yarns exhibit less sagging than APIC, as they stretch less under their own weight. The reduced damping also makes the PolyPIC yarns spring back from being stretched by the fluid faster.
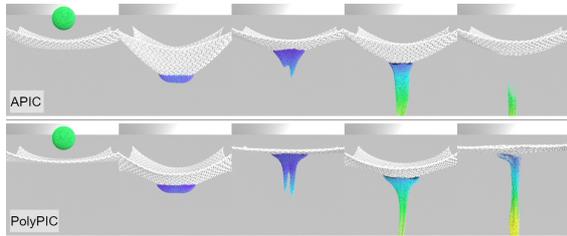

Figure 6: Fluid splashing onto a small square of yarn fabric using APIC (top) and PolyPIC (bottom) transfers. Particle colours indicate velocity (dark blue = low, red = high). The PolyPIC yarns exhibit less sagging than APIC (see frame 1). The reduced damping makes the PolyPIC yarns spring back faster from being stretched (see frames 3-5).

Finally, a ball of fluid falling onto a medium square of yarn fabric is shown in Figure 7. This example demonstrates that PolyPIC increases the transfer of energy from the falling fluid to the suspended yarn. Due to issues with collision detection and resolution, this increased transfer of energy causes the yarns to become more tangled than when using APIC, as can be seen in the later frames. This scenario is also shown in Figure 1 as a fully rendered sequence. The surface reconstruction was performed using SideFx's Houdini 3D graphics software (SideFX, 2023). A more detailed description of the rendering process can be found in section 4 of the supplemental material of Fei et al.
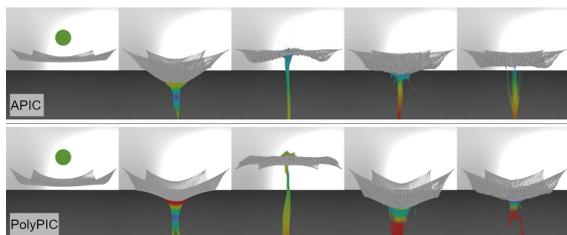

Figure 7: Fluid splashing onto a medium sized square of yarn fabric using APIC (top) and PolyPIC (bottom) transfers. Particle colours indicate velocity (dark blue = low, red = high). PolyPIC causes more energy to be transferred from the falling fluid to the yarn and results in a more dynamic result.

A breakdown of performance information for each presented example is shown in Table 2. Particles make up the objects that are being simulated. In the simulation, all particles are flagged as being either fluid or solid particles. Fluid particles are the particles used to simulate the bulk fluid. Elements are the mesh elements that make up the fabric (triangles for cloth simulations, rod segments for yarn simulations) and are used for calculating the movement of absorbed fluids. The 'small' simulations use a piece of fabric that is ¼ the size of the 'medium' simulation (½ of both the width and height). Simulations were run using an Intel i9-12900K CPU, and each example was repeated 3 times and the mean performance results are presented. The source code can be found at https://github.com/robden820/libwetcloth. For simulations involving PolyPIC, 8 fluid scalar modes and 8 solid scalar modes were used for all examples.

As can be seen in Table 2, as the number of particles increases, the impact of the additional complexity of the PolyPIC model increases. The dam break scenarios use the largest number of particles to clearly demonstrate the difference in fluid behaviours when using PolyPIC, but also to highlight the additional computational impact of PolyPIC over APIC. The small cloth splash scenario has the smallest number of simulated particles and elements and shows the smallest difference in seconds per step and peak memory usage between APIC and PolyPIC of all the examples. Some of the simulations require use of a smaller timestep value due to stability issues (see Section 5).

Additionally, an energy plot is presented for the dam break scenario in Figure 8 and for the interactions scenarios in Figure 9 (due to variation in the simulation setup, each simulation stabilises around a different final mean energy). The plots show the mean energy per particle for the presented scenarios, calculated as the sum of the kinetic energy and gravitational potential energy. Figure 8 shows that for a simulation involving only fluid particles, the improvement in energy preservation gained by using PolyPIC in the place of APIC is minimal, however, as shown in Figure 4 there is still a difference in the simulations visual output achieved using PolyPIC. Figure 9 shows a similar trend for the interaction scenarios, and PolyPIC has a small impact in the long term energy preservation of the simulation. However, simulations involving PolyPIC require more time to reach a stable energy level. The increased oscillations of PolyPIC before reaching this stable energy level indicate that PolyPIC suffers less from numerical damping than APIC, leading to more dynamic simulations.

5 DISCUSSION

As shown in the comparison scenarios, visually, PolyPIC provides a notable increase in simulation dy-

Table 2: Timing and storage data for APIC and PolyPIC for three example scenarios. [†] The dam break simulations involved no fabric, so required 0 mesh/rod elements.

| Example			simulation duration (s)	timestep (s)	s/step (avg)	total run time (mins)	peak memory (GB)	#particles (avg)	#fluid particles (avg)	#elements (avg)
Dam Break Small (APIC)			5.0	0.0002	0.510	217	1.978	310929	310929	0[†]
Dam Break Small (PolyPIC)			5.0	0.0002	1.172	499	2.040	310839	310839	0[†]
Splash Cloth Small (APIC)			4.0	0.0002	0.480	167	0.896	12359	1360	20406
Splash Cloth Small (PolyPIC)			4.0	0.0001	0.428	282	0.842	12228	1228	20406
Splash Yarn Small (APIC)			4.0	0.0002	0.454	155	1.056	23074	3471	19799
Splash Yarn Small (PolyPIC)			4.0	0.0002	0.579	197	1.109	23233	3630	19799
Splash Yarn Medium (APIC)			4.0	0.0002	1.897	818	9.300	119520	40320	79596
Splash Yarn Medium (PolyPIC)			4.0	0.00005	1.460	1958	5.764	116416	37213	79596
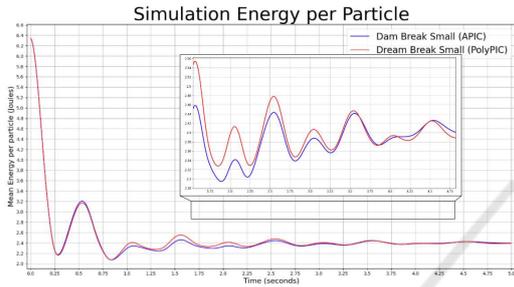
Figure 8: Mean particle energy for the dam break scenario, calculated as the sum of kinetic energy and gravitational potential energy. PolyPIC improves the energy preservation over APIC, but the effect per particle is minor.
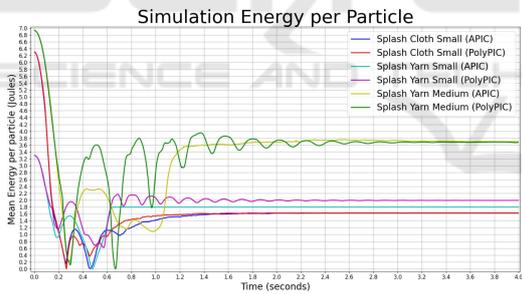

Figure 9: Mean particle energy for the interaction scenarios, calculated as the sum of kinetic energy and gravitational potential energy. Change in energy preservation in PolyPIC and APIC is minimal, although the energy of PolyPIC oscillates for a longer period before reaching a constant value. Due to differences in the simulation setup (e.g. fabric at different world heights, rigid sphere in cloth scenario), each simulation stabilises around a different final mean energy.

namics. Whilst theoretically PolyPIC has been shown to be lossless when transferring velocity between the particles and the grid (shown in the supplementary material of Fu et al), this can't be achieved in practice. For fluids, using polynomial modes involving multi-quadratic terms (i.e. $N_r > 2^d$) causes the simulation to become unstable, even when a very small time step is used. For solids in a non-coupled sce-

nario, the maximum number of modes can be used without effecting the overall stability of the simulation. However, when simulating coupled interaction, stability issues become increasingly severe as higher order modes are introduced.

As can be seen in Table 2, when comparing the dam break scenarios, an increase in the time required for each simulation step can be seen when using PolyPIC rather than APIC. In the interaction scenarios, the time per simulation generally decreases, but the average number of fluid particles in the simulation also decreases. The reduced numerical damping of PolyPIC causes the fluid to splash off the fabric rather than being absorbed, so the fluid particles leave the domain of the simulation faster, and the particles are deleted from the simulation. This reduces the number of fluid particles to be simulated in later timesteps, allowing each step to be simulated faster.

The fabric-interaction simulations for APIC were both able to be simulated using a timestep of $\Delta t = 2e^{-4}s$, but using this value for the PolyPIC scenarios caused the simulation to become unstable. Whilst the same timestep enabled stable simulation of the small yarn PolyPIC scenario, we found using a timestep of $\Delta t = 1e^{-4}s$ and $\Delta t = 5e^{-5}s$ allowed the small cloth and medium yarn PolyPIC simulation to remain stable for their duration, respectively. This means that while in general there was only a small change in the time required for each simulation step, the need for a smaller timestep for PolyPIC resulted in an overall increase in the time taken to run the simulation, as shown in Table 2.

Possible future work would be to experiment with more stable simulation frameworks such as position-based dynamics (Müller et al., 2007) which has been shown to reduce the need for small timesteps. Also, in the yarn examples, the individual yarns quickly become tangled after collision with the fluid. Using a more robust collision handling technique such as incremental potential contact (Li et al., 2020) could improve the stability of the simulations.

6 CONCLUSIONS

The work presented by Fei et al is widely considered to be the state-of-the-art for liquid-fabric interaction simulations. This paper has replaced APIC with PolyPIC in their model and performed a comparison of the PolyPIC model with the APIC model in the context of liquid-fabric interactions. Using PolyPIC in place of APIC for liquid-fabric interaction improves the dynamism of simulations, increasing energy transfer between the fluid and cloth/yarn and increasing the resolution of vortical details and small scale splashes. However, PolyPIC has a higher computation cost over APIC and the reduced numerical damping of PolyPIC also caused stability issues requiring the use of a smaller time step. This requirement for smaller timesteps highlights the need for a greater consideration of techniques to improve simulation stability and inter-yarn collision detection. Despite this limitation, this paper demonstrates the potential of PolyPIC as a method of improving liquid-fabric interaction simulations based on PIC methods.

ACKNOWLEDGEMENTS

REFERENCES

Bergou, M., Wardetzky, M., Robinson, S., Audoly, B., and Grinspun, E. (2008). Discrete elastic rods. *ACM Transactions on Graphics*, 27(3):1–12.

Brackbill, J. and Ruppel, H. (1986). FLIP: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions. *Journal of Computational Physics*, 65:314–343.

Fang, Y., Li, M., Gao, M., and Jiang, C. (2019). Silly rubber: an implicit material point method for simulating non-equilibrated viscoelastic and elastoplastic solids. *ACM Transactions on Graphics*, 38:1–13.

Fei, Y. R., Batty, C., Grinspun, E., and Zheng, C. (2018). A multi-scale model for simulating liquid-fabric interactions. *ACM Trans. Graph.*, 37(4).

Fu, C., Guo, Q., Gast, T., Jiang, C., and Teran, J. (2017). A polynomial particle-in-cell method. *ACM Transactions on Graphics*, 36(6):222:1–222:12.

Grinspun, E., Hirani, A., Desbrun, M., and Schröder, P. (2003). Discrete Shells. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*.

Harada, T., Koshizuka, S., and Kawaguchi, Y. (2007). Real-time Fluid Simulation Coupled with Cloth. *EG UK Theory and Practice of Computer Graphics*.

Harlow, F. H. (1962). The particle-in-cell method for numerical solution of problems in fluid dynamics. Technical Report LADC-5288.

Huber, M., Pabst, S., and Straßer, W. (2011). Wet cloth simulation. In *International Conference on Computer Graphics and Interactive Techniques*.

Jiang, C., Schroeder, C., Selle, A., Teran, J., and Stomakhin, A. (2015). The affine particle-in-cell method. *ACM Transactions on Graphics*, 34(4):51:1–51:10.

Lenaerts, T., Adams, B., and Dutré, P. (2008). Porous flow in particle-based fluid simulation. *ACM Trans. Graph.*, 27.

Li, M., Ferguson, Z., Schneider, T., Langlois, T., Zorin, D., Panozzo, D., Jiang, C., and Kaufman, D. M. (2020). Incremental potential contact: intersection- and inversion-free, large-deformation dynamics. *ACM Transactions on Graphics*, 39(4).

Müller, M., Heidelberger, B., Hennix, M., and Ratcliff, J. (2007). Position based dynamics. *Journal of Visual Communication and Image Representation*, 18(2):109–118.

Nielsen, M. B. and Østerby, O. (2013). A two-continua approach to Eulerian simulation of water spray. *ACM Transactions on Graphics*, 32(4):1–10.

Patkar, S. and Chaudhuri, P. (2013). Wetting of Porous Solids. *IEEE Transactions on Visualization and Computer Graphics*, 19(9):1592–1604.

Shao, X., Wu, W., and Wang, B. (2018). Position-based simulation of cloth wetting phenomena. *Computer Animation and Virtual Worlds*, 29(1):e1788.

SideFX (2023). Houdini - 3D modeling, animation, VFX, look development, lighting and rendering | SideFX.

Stomakhin, A., Schroeder, C., Chai, L., Teran, J., and Selle, A. (2013). A material point method for snow simulation. *ACM Transactions on Graphics*, 32(4):1–10.

Stomakhin, A., Schroeder, C., Jiang, C., Chai, L., Teran, J., and Selle, A. (2014). Augmented MPM for phase-change and varied materials. *ACM Transactions on Graphics*, 33(4):1–11.

Sulsky, D., Chen, Z., and Schreyer, H. L. (1994). A particle method for history-dependent materials. *Computer Methods in Applied Mechanics and Engineering*, 118(1):179–196.

Zheng, Y., Chi, X., Chen, Y., and Wu, E. (2021). Stains on imperfect textile. *Virtual Reality & Intelligent Hardware*, 3:142–155.