

Particle-based simulation of the interaction between fluid and knitwear

Uğur Güdükbay · Serkan Bayraktar · Çetin Koca ·
Bülent Özgüç

Received: 23 June 2011 / Revised: 27 January 2012 / Accepted: 27 March 2012 / Published online: 19 April 2012
© Springer-Verlag London Limited 2012

Abstract We present a particle-based method to simulate and visualize the interaction of knitwear with fluids. The knitwear is modeled using spring-mass systems and the fluid is modeled using the smoothed particle hydrodynamics method. Two-way coupling is achieved by considering surface tension, capillary, and interparticle forces between the fluid and knitwear. The simulation of fluid and knitwear particles is performed on the graphics processing unit. Photorealistic rendering of knitwear and fluid is achieved by using a hardware-accelerated rasterization-based rendering technique. Our method is able to simulate and visualize the macro- and microstructure of free-form knitwear and reflective and refractive characteristics of the fluid surface.

Keywords Physically-based modeling · Fluid simulation · Particle-based modeling · Smoothed particle hydrodynamics · Knitwear · Graphics processing unit · Lumislice

Electronic supplementary material The online version of this article (doi:[10.1007/s11760-012-0308-2](https://doi.org/10.1007/s11760-012-0308-2)) contains supplementary material, which is available to authorized users.

U. Güdükbay (✉) · S. Bayraktar · Ç. Koca · B.Özgüç
Department of Computer Engineering, Bilkent University,
Bilkent, 06800 Ankara, Turkey
e-mail: gudukbay@cs.bilkent.edu.tr

S. Bayraktar
e-mail: serkan.bayraktar@uni-siegen.de

Ç. Koca
e-mail: ckoca@cs.bilkent.edu.tr

B.Özgüç
e-mail: ozguc@bilkent.edu.tr

1 Introduction

Modeling and simulating of fluid and cloth and their interaction is a popular research topic in computer graphics. Particle-based systems are commonly used since they are easy to model and versatile enough to define behavior through manipulating particle-to-particle interaction. We propose a particle-based simulation method for modeling knitwear and its interaction with fluid. Our knitwear model is based on the mass-spring method. A mass-spring mesh defines the underlying structure of the knitwear, and three layers of mass points are connected by volume-preserving spring constraints to model the thickness of the knitwear. We use a volumetric rendering method, which is based on hardware rasterization, to render the knitwear. Thread points that constitute the skeleton of the knitwear rendering process are kinematically controlled by interpolating the mass point coordinates. Our fluid model uses the smoothed particle hydrodynamics (SPH) for the simulation and the Marching Cubes algorithm for the fluid surface rendering. The two-way coupling of knitwear and fluid is represented by the discrete element method (DEM) and forces resulting from the capillary pressure. Photorealistic rendering of knitwear is a challenging task since the microstructure of threads should be captured to portray the fluffy characteristic of the yarn. The problem becomes more complicated when free-form knitwear must be visualized from different angles. We use a hardware-accelerated volumetric rendering technique that is based on the lumislice method proposed by Chen et al. [1]. To increase realism, we render soft shadows by the cascaded shadow mapping technique [2]. To speed up the simulation and rendering, we implement the method on the GPU, which is suitable for parallel tasks. The simulation of the interaction between knitwear and fluid is performed using CUDA, which provides a convenient interface for

high-level numerical computation; rendering is handled by GLSL.

The main contribution of this paper is a practical, particle-based method to simulate a volumetric knitwear model and its interaction with fluids, including absorption and self-collisions. Our model provides a convenient basis for knitwear rendering and defining object interactions in a unified particle system. We also propose several improvements over the existing methods to provide a practical way for rendering knitwear and fluid together. Specifically, we address the issues arising from a texture-based rendering of knitwear, combining soft shadows with rasterization-based reflections and refractions, and simulating the thread structure of the knitwear.

The paper is organized as follows: In Sect. 2, we overview the state of the art on particle-based fluid modeling and the approaches for modeling, simulating and rendering knitwear. Section 3 describes the modeling and simulation of knitwear. Section 4 discusses the simulation of fluids. Section 5 presents the proposed method for simulating the interaction between knitwear and fluid. Section 6 discusses the rendering of knitwear and fluid, and the multi-pass rendering of the scene. Section 7 gives the results. Section 8 concludes the paper.

2 Related work

Particle systems have been popular for modeling and simulating natural phenomena, such as fluids, fire, smoke, cloth, and woven fabrics. Miller et al. [3] utilize particle interactions for viscous fluid simulation. Terzopoulos et al. [4] use a particle system to model melting objects. They connect particles with spring constraints and modify spring constants whenever the object changes its phase. Desbrun et al. [5] propose a SPH-based [6] method for simulating highly deformable objects. Müller et al. [7] use SPH to simulate incompressible fluids. Solenthaler et al. [8] propose a method based on SPH and an elastic–plastic model to simulate fluid, deformable bodies, melting and solidification. Hong et al. [9] combine an Eulerian grid-based fluid simulator with the SPH method to mimic realistic bubble and foam behavior. Lasasso et al. [10] couple the particle level set method and SPH to simulate large fluid bodies with diffuse regions. Coupling of SPH-based fluid simulation and rigid bodies has been recently investigated in several papers. Becker et al. [11] propose a direct forcing schema to enforce slip and Neumann boundary conditions, drag effects, and buoyancy. Ihmsen et al. [12] propose a boundary handling schema.

In computer graphics, woven fabric and knitwear are usually modeled and simulated by particles where spring-mass systems are employed to simulate deformations. Eberhardt et al. [13] use a particle-based system to simulate the behavior

of woven and knitted cloth. Nocent et al. [14] use a spline-based model to simulate a cloth plane and project the deformations into yarn control vertices. Chen et al. [1, 15] exploit the repetitive structure of knitted cloth to simulate and render knitwear. They use spring forces and a force field model to produce a realistic animation. To render knitted cloth, they use the lumislice method, where they define lumislice as a cross-section of yarn, and use it to determine the radiance from a yarn cross-section. Kaldor et al. [16] model each yarn as an inextensible, yet otherwise flexible, B-spline tube. Stiff penalty forces and rigid-body velocity filters are used to simulate knitted cloth behavior.

Gröller et al. [17] employ a volume-rendering technique to render the yarn microstructure of the knitwear efficiently. Their technique, however, can only handle knitwear on a flat surface. The curved ray-tracing method proposed in [18] has a very high cost of rendering and it is very difficult to create complex knitwear stitching patterns with it. The Gouraud-shaded triangle-based method proposed in [1] is more efficient but does not handle close-up views of knitwear. Durupinar et al. [19] propose a procedural method for visualizing knitted fabrics; however, it cannot handle complex stitch patterns but works in real time due to the simplicity.

3 Modeling and simulating knitwear

Mass-spring networks are the most widely used method in cloth simulation. They are simple to implement and easy to fine tune for desired effects. In mass-spring models, discrete mass points are connected by damped springs in several combinations to mimic cloth's behavior in the presence of stretching, bending, and shearing forces [20]. Knitwear behavior is similar to that of a thin cloth, with the exception of a more prominent thickness. To achieve volumetric representation of knitwear without sacrificing the simplicity and speed of the mass-spring method, we use three layers of mass-spring meshes connected by volumetric springs. Each layer is modeled by shearing, stretching, and bending springs (Fig. 1a), and the whole structure provides a dynamic thickness so that it reacts to perpendicular forces in a realistic way.

Self-collision of the knitwear is handled on a particle-to-particle basis, where penalty forces are computed by the DEM. Particle proximities are detected by a grid-based neighbor search algorithm, where mass points connected by springs are ignored. The thread structure of knitwear is constructed by stitches that consist of bonding points, whose coordinates are computed by interpolating the neighboring mass points of the mesh. To exploit the three-layered mass point structure and give a volumetric appearance to the simulated knitwear, some of the bonding points are interpolated by using the mass point coordinates of layer three and some

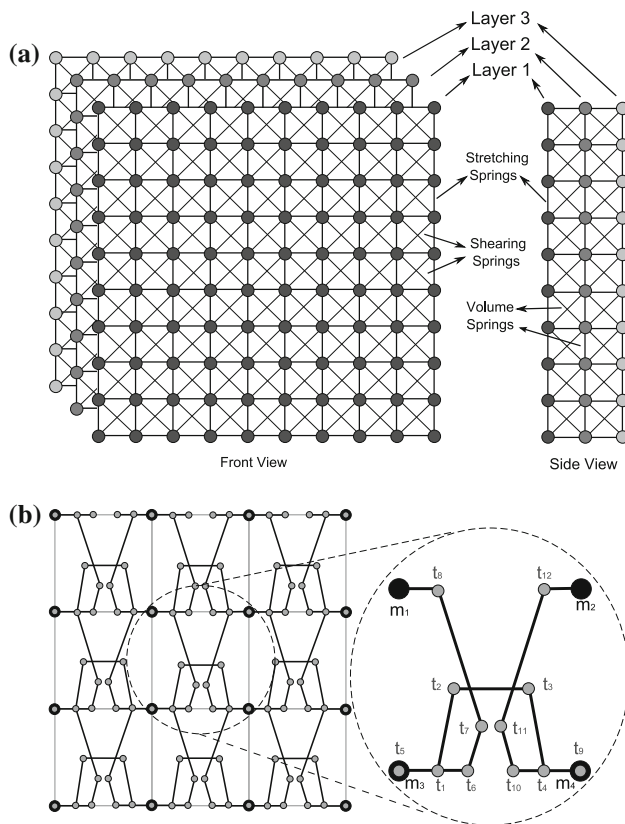


Fig. 1 **a** Mass-spring structure of our knitwear model. To simulate the thickness of knitwear, the layers are connected by volume-preserving springs. **b** Bonding points (gray dots) and mass points (black dots)

of the stitch control points are interpolated by using the mass point coordinates of layer one.

At each time step, mass points are moved as a result of the simulated external forces (e.g., gravity, wind, collision, user forces) and internal forces (e.g., stretching, bending and shearing forces), and the bonding points are updated accordingly (see Fig. 1b). In Fig. 1b, the gray and black dots are bonding points and mass points, respectively, and the gray lines represent springs. The interpolation equations for the positions of stitch control points are similar to those described in [19], except that we exploit the layered structure of the mass-spring mesh to model thickness. Equation 1 describes the interpolation of the positions of the bonding points for the thread structure depicted in Fig. 1b. In Eq. 1, m_i s and m'_i s are mass points that belong to layers one and three, respectively.

$$\begin{aligned}
 t_1 &= \frac{6}{7}m_3 + \frac{1}{7}m_4 \\
 t_2 &= \frac{5.5}{21}m'_1 + \frac{1.5}{21}m'_2 + \frac{11}{21}m'_3 + \frac{3}{21}m'_4 \\
 t_3 &= \frac{1.5}{21}m'_1 + \frac{5.5}{21}m'_2 + \frac{3}{21}m'_3 + \frac{11}{21}m'_4 \\
 t_4 &= \frac{1}{7}m_3 + \frac{6}{7}m_4
 \end{aligned}$$

$$\begin{aligned}
 t_5 &= m'_3 \\
 t_6 &= \frac{5}{7}m'_3 + \frac{2}{7}m'_4 \\
 t_7 &= \frac{4.5}{28}m_1 + \frac{2.5}{28}m_2 + \frac{13.5}{28}m_3 + \frac{7.5}{28}m_4 \\
 t_8 &= \frac{6}{7}m_1 + \frac{1}{7}m_2 \\
 t_9 &= m'_4 \\
 t_{10} &= \frac{2}{7}m'_3 + \frac{5}{7}m'_4 \\
 t_{11} &= \frac{2.5}{28}m_1 + \frac{4.5}{28}m_2 + \frac{7.5}{28}m_3 + \frac{13.5}{28}m_4 \\
 t_{12} &= \frac{1}{7}m_1 + \frac{6}{7}m_2
 \end{aligned} \quad (1)$$

4 Simulating fluid

In computational physics, fluid flow is generally defined by the Navier–Stokes momentum equation, which is in form of: $\rho \left(\frac{\partial \mathbf{v}}{\partial t} \right) = -\nabla p + \rho \mathbf{f} + \mu \nabla^2 \mathbf{v}$, where ρ is the fluid's density, $\mathbf{f}$ is the sum of external forces, and p is the pressure.

We implement the Navier–Stokes momentum equation by using the SPH [6]. Originally developed to be used in astrophysics and computational physics, SPH has been used in computer graphics to simulate fluid, deformable bodies, and granular materials. SPH is an Lagrangian method that defines the fluid body by a set of discrete particles, p_i . The parameters defining the characteristics of fluid, that is, mass, density, viscosity, pressure, and surface normal, are defined for each particle. These parameters are distributed around the particles according to an interpolation function, which is called kernel and denoted by W . The kernel defines a sphere with a radius h (kernel radius) around each particle. Fluid parameters in arbitrary positions are computed by interpolating the parameters of neighboring particles.

The density ρ_i of particle i position is computed by:

$$\rho_i = \sum_j m_j W(\mathbf{r}_{ij}, h), \quad (2)$$

where $\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j$, m_j is the mass of particle j , and $\mathbf{r}_i$ is particle i 's position.

The pressure P_i of particle i is computed by $P_i = k(\rho_i - \rho_0)$, where ρ_0 is the global rest density, and k is the scaling coefficient. The pressure force $\mathbf{F}_i^{\text{pressure}}$ exerted on particle i is computed by:

$$\mathbf{F}_i^{\text{pressure}} = - \sum_j m_j \left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2} \right) \nabla W(\mathbf{r}_{ij}, h). \quad (3)$$

The viscous force, $\mathbf{F}_i^{\text{viscous}}$, acting on particle i is computed by:

$$\mathbf{F}_i^{\text{viscous}} = \sum_j \frac{m_j(\mu_i + \mu_j)\mathbf{V}_{ij}}{\rho_i \rho_j} \left(\frac{1}{\|\mathbf{r}_{ij}\|} \frac{\partial W_{ij}}{\partial \mathbf{r}_i} \right), \quad (4)$$

where $\mathbf{V}_{ij} = \mathbf{V}_i - \mathbf{V}_j$ is the relative velocity of particle i with respect to particle j , and μ_i is the viscosity of i .

For density and force computations, we choose to use the spiky kernel function [21] defined as:

$$W(\mathbf{r}, h) = \frac{15}{\pi h^6} \begin{cases} (h - r)^3 & \text{if } 0 \leq |\mathbf{r}| \leq h \\ 0 & \text{if } |\mathbf{r}| \geq h, \end{cases} \quad (5)$$

since it does not cause particle clustering and it has vanishing first and second derivatives at the boundary. The gradient of the spiky kernel is:

$$\nabla W(\mathbf{r}, h) = -\frac{45}{\pi h^6} \begin{cases} (h - r)^2 \mathbf{r} & \text{if } 0 \leq |\mathbf{r}| \leq h \\ 0 & \text{if } |\mathbf{r}| \geq h. \end{cases} \quad (6)$$

5 Simulating knitwear–fluid interaction

Interactions between knitwear and fluid and knitwear itself are defined on a particle-to-particle basis. We can categorize the forces that knitwear and fluid particles exert on each other into two classes. The first class is due to particle proximity, which is modeled by the DEM [22]. The second class acts on fluid particles, due to the porous structure of knitwear. These forces are defined by the capillary pressure differences.

5.1 Discrete element method (DEM)

The discrete element method is a popular numerical method for defining interparticle forces, especially in granular materials. The contact force $\mathbf{F}_{i,j}$ between two particles i and j can be computed as the sum of tangential and normal components: $\mathbf{F}_{i,j} = \mathbf{F}_{i,j}^n + \mathbf{F}_{i,j}^t$. The normal force component $\mathbf{F}_{i,j}^n$ is repulsive and acts on particle centers in the direction of the vector $\mathbf{r}_{i,j} = \mathbf{r}_i - \mathbf{r}_j$, where $\mathbf{r}_i$ and $\mathbf{r}_j$ are the centers of particles i and j , respectively. $\mathbf{F}_{i,j}^n$ is computed according to Eq. 7:

$$\mathbf{F}_{i,j}^n = k \frac{\mathbf{r}_{i,j}}{\|\mathbf{r}_{i,j}\|} u_{i,j} - c \left(\mathbf{V}_i^n - \mathbf{V}_j^n \right), \quad (7)$$

where k is the compression stiffness coefficient, c is the coefficient of viscous damping, $\mathbf{V}_i^n$ is the velocity component of particle i in the direction of $\mathbf{r}_{i,j}$, and $u_{i,j} = \|\mathbf{r}_{i,j}\| - (\text{radius}_i + \text{radius}_j)$ is the particle overlap.

$\mathbf{F}_{i,j}^t$ is the tangential shear force and defines the resistance to the movement that particles exert on each other in the tangential direction:

$$\mathbf{F}_{i,j}^t = \mu (k u_{i,j}) \frac{\mathbf{V}_i^t - \mathbf{V}_j^t}{\|\mathbf{V}_i^t - \mathbf{V}_j^t\|}, \quad (8)$$

where k and $u_{i,j}$ are as defined above, $\mathbf{V}_i^t$ and $\mathbf{V}_j^t$ are the tangential velocities of particles i and j , respectively, and μ is the friction coefficient. This formulation of tangential shear force scales with the magnitude of normal force, thus ensuring stability.

5.2 Capillary pressure

The porous structure of knitwear exerts forces on neighboring fluid particles that move within the material toward less-saturated regions. Two main defining characteristics of a porous material are porosity (denoted by ϕ) and permeability (denoted by κ) [23]. Porosity is the fraction of material's void volume to its total volume. It represents the porous material's fluid absorption capacity. Saturation can be defined as the ratio of total fluid volume the porous medium can hold to the current absorbed fluid volume. Hence, one can define saturation by:

$$s = \frac{m_{\text{fluid}}}{\phi V \rho_{\text{fluid}}}, \quad (9)$$

where V is the total volume of the material, ρ_{fluid} is the fluid density, and m_{fluid} is the fluid mass. Permeability is the ability of a porous material to transmit fluids and is mainly controlled by the size and interconnectivity of pores within the material. For isotropic materials, permeability can be represented by a scalar.

The main force acting on a fluid particle inside a porous material is due to capillary pressure [24], which is the pressure difference in multiphase flows that occurs across the interface. Capillary pressure can directly be expressed by the porous material's saturation and can be defined as:

$$P_{\text{capillary}}(s) = \frac{\sigma J(s)}{\sqrt{\kappa/\phi}}, \quad (10)$$

where s is saturation, κ is permeability of the material, ϕ is porosity, and σ is a coefficient to control the pressure. $J(s)$ is the Leverett function that mainly depends on the morphology of the porous material. We use the following Leverett function [25] to define hydrophilic materials:

$$J(s) = 1.417(1 - s) - 2.120(1 - s)^2 + 1.263(1 - s)^3. \quad (11)$$

To define porous characteristics for our particle-based knitwear model, we associate each mass point i with porosity ϕ_i and volume V_i [23]. The total capillary force acting on a fluid particle j , which is exerted by the neighboring knitwear mass points i , can be computed by:

$$\mathbf{F}_{\text{capillary}}^j = \sum_i P_{\text{capillary}}^i \nabla W(\mathbf{r}_{ij}, h), \quad (12)$$

where $\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j$, and $\mathbf{r}_i$ and $\mathbf{r}_j$ are the positions of neighboring mass point i and fluid particle j , respectively.

The additional weight of fluid penetrated into knitwear is computed by the interpolation $m_{\text{fluid}} = \sum_j m_j W$, where m_j is the mass of neighboring fluid particles, and W is the smoothing kernel.

6 Rendering

6.1 Rendering knitwear

Rendering the knitwear strand is important because the rendering process needs to synthesize the microstructure of the knitwear, which consists of a huge number of thin fibers and has a fuzzy look. We use the lumislice primitive [1] as a 2D texture, which represents the distribution of the yarn fibers in a cross-section of the yarn.

Using the alpha-blending capabilities of the graphics hardware, we render a multi-dimensional array of quads textured with a 2D lumislice to obtain a volumetric rendering. The problem with this approach is that the alpha-blended one-dimensional array of textured quads does not handle view angles that are non-perpendicular to rendered quads. This effect is illustrated in Fig. 2a. To overcome this problem, instead of the one-dimensional array of textured quads, we construct a three-dimensional grid of voxels for volumetric rendering. In this approach, there is a one-dimensional array of quads along each coordinate axis of the local coordinate system. Thus, even if one or two of the one-dimensional quad arrays are parallel to the viewing direction, it is guaranteed that the last one is not. Fig. 2b illustrates the result.

One limitation of the hardware-accelerated alpha-blending is that the quads must be rendered in back-to-front order because of the way blending works in hardware: the rendering pipeline must know the pixel at the back when rendering the pixel at the front to be able to compute the correct color of the final pixel. This sorting operation must be done whenever

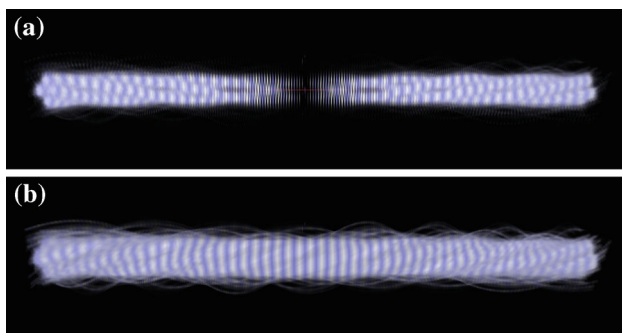


Fig. 2 **a** The problem with one-dimensional array of two-dimensional quads is shown. This effect occurs when the quads lie parallel to the viewing direction. **b** The problem is solved by using a 3D grid of voxels as explained in the text

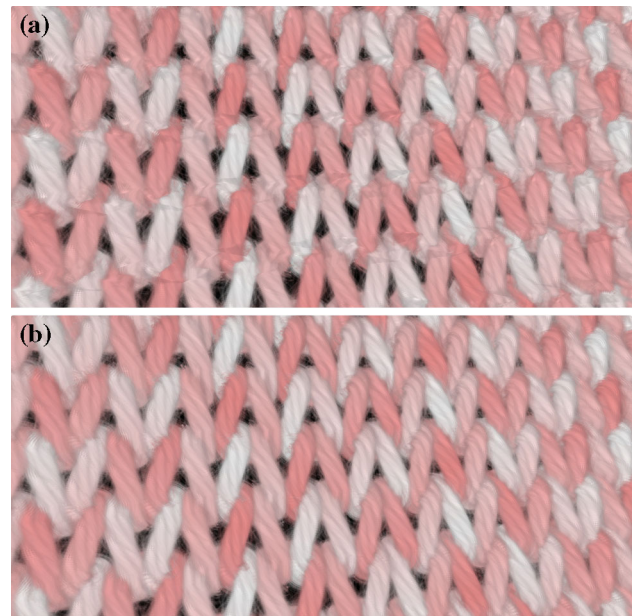


Fig. 3 **a** The effect of discontinuities in the overlaps at the segment joints. **b** The effect is alleviated by fitting a Catmull–Rom spline on the bonding points

the viewer orientation or model position/orientation changes. To sort the quads, we employ the bitonic sort on the GPU.

After sorting, we twist the grid of quads along the axis defined by the normal of each quad to increase the visual quality of the strand. The 3D grid of voxels is aligned along the path defined by the stitch bonding points. When the voxels are positioned on the straight line between the bonding points, visual artifacts appear because of the overlaps at the joints, as illustrated in Fig. 3a. To overcome this problem, we fit a Catmull–Rom spline on the bonding points, which is used to determine the path of each thread of the knitwear. This provides a smoother path for the quads and consequently results in much better visual quality. The tangents of the Catmull–Rom spline at specific points are used to compute the normals of the corresponding quads. In this improved approach, instead of rendering a different yarn segment corresponding to each line segment, we render the whole yarn thread at once by placing the quads at regular intervals on the spline. Rendered this way, no discontinuity results in a single thread, and thus, the image quality is much higher, as shown in Fig. 3b.

Soft shadows are indispensable for a realistic, high-quality rendering of a knitwear model; they help stitches stand out, especially when the cloth is folded or there are many layers of cloth on top of each other. To achieve a high-quality rendering of soft shadows, we employ a cascaded shadow mapping technique [2] that divides the frustum into a number of segments, making it possible to use different texture resolutions and obtain a high level of antialiasing. As Fig. 4 illustrates, using cascaded shadows improves the visual quality of our knitwear model.

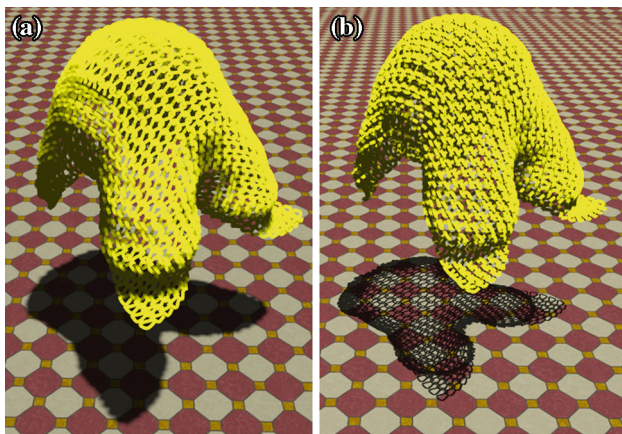


Fig. 4 Shadows rendered with (a) traditional shadow mapping and (b) cascaded shadow mapping

6.2 Rendering fluid

To achieve a uniform visualization of a scene containing knitwear and fluid, we choose a hardware-accelerated, rasterization-based rendering approach for the fluid surface rendering. Our approach visualizes the polygonized fluid surface by adding reflections and refractions. Polygonization of the fluid surface is achieved by the Marching Cubes algorithm, which computes an isosurface on a fine uniform grid. When naively applied, this algorithm produces a bumpy surface in flat regions and fails to capture fine details in regions like drops and waves. We improve surface generation by considering the relative particle positions with respect to the fluid surface so that particles close to the surface contribute less to the computed isovalue [26].

Reflections on the surface are approximated by an environment-mapped cube approach where the scene is rendered into six textures, excluding the reflective object (fluid, in this case). For each fragment of the fluid surface, the fragment normal is traced to find the corresponding pixel on the environment map. For dynamic scenes, the environment map must be created at each time step. Refractions due to the fluid surface are computed by Snell's law by using fragment normals, the ray from the eye position to the fragment, and the texture. For a faster computation of scalar values for each vortex of the Marching Cubes grid, we employ a sorting-based neighbor search method [27] to determine the vortex's neighboring particles more efficiently.

6.3 Rendering partially wet knitwear

An obvious fact in rendering the interaction of knitwear and fluid is that the portions of the knitwear that touch fluid get wet, and thus, we need to simulate the saturation of the knitwear and render wet portions of the knitwear in a different way. In our approach, we basically use an ad-hoc method to

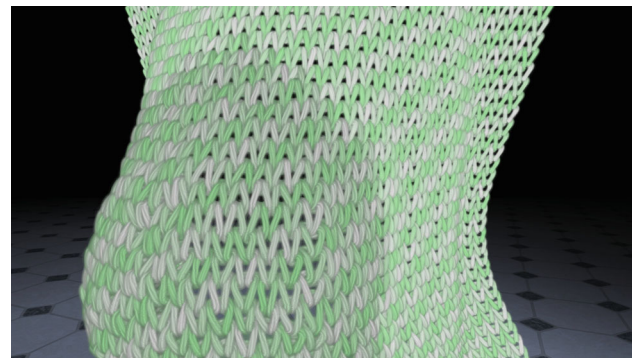


Fig. 5 A partially wet knitwear

change the color of the wet portions of the knitwear: $C' = C(1.0 - \alpha\sqrt{s})$, where C is the original color of the knitwear, s is the saturation as defined in Eq. 9, α is a constant, and C' is the final color. To speed up the rendering process, we use a vertex-based approach instead of a pixel-based approach. Our experiments have shown that the value $\alpha = 0.15$ produces good results for the wetness effect. An improvement to this technique is that we also slightly increase the strand radius where the strand is wet, since the fluid is absorbed into these portions of knitwear. Figure 5 shows a partially wet knitwear where the central part of the knitwear is wet.

6.4 Rendering the scene

The proposed method performs several rendering passes to achieve a uniform rendering of knitwear and fluid. The first stage of our rendering algorithm creates the shadow maps in several steps, with one step for each shadow map. For each shadow map, the viewport and light position are updated according to the shadow map's coverage area. We need to transform the knitwear skeleton pivots and generate the knitwear model to accommodate the change in the viewport. The final step for creating the shadow maps is to depth sort the knitwear points and render the final image for depth information since that data are needed for generating the shadow maps.

The next stage generates reflection maps. For each side of the environment-map cube, we update the viewpoint and render the scene into a texture. We do not render the fluid in this step, since it should not reflect itself. Next, we perform two additional rendering passes for the refraction map. In the first pass, we collect the depth information of the refractive fluid surface. In the second pass, we render the scene, excluding the refractive surface. We omit fragments that are closer to the viewer than to the refractive surface since we do not want the refractive surface to refract what is in front of it. We use the depth data computed in the first pass to be able to do this.

We use reflection textures to texture the environment map, shadow maps to decide whether a fragment is in shadow, and refraction maps to compute the final color of a refracted ray. To render the final image, we first render the static environment in the scene. Then, we render the fluid surface using reflection, refraction, and shadow maps. Finally, we render the knitwear using only shadow maps and then render the final image. One issue with this approach is that since we compute the reflection maps before the refraction maps the refractions are not visible in the reflections on the fluid surface. In our observations, this was not a significant problem and most of the time it was not even visible to the viewer.

7 Results

This section presents sample simulations where knitwear and fluid are simulated and rendered by the proposed method. The method is implemented on GPU by CUDA and GLSL. The velocity Verlet method is used as the

numerical integrator. A sorting-based neighbor search algorithm is employed for the fast lookup of particle neighbors and the fast isovalue computation for the Marching Cubes algorithm.

Figure 6 exhibits still images from a simulation where knitwear drops onto a rigid object. The images show the effect of adding soft shadows and texturing. Figure 7 shows still frames from the final rendering of the simulation of fluid and knitwear. The simulation of the scene (determining particle neighborhood information, computing fluid pressure-based forces, computing two-way coupling forces, and interpolating knitwear control point positions) takes about 1.2 s (with 64K fluid and 50K knitwear particles). The knitwear is rendered using volumetric rendering techniques; the figure also shows the effect of the knitwear getting partially wet. The scene consists of about 13 million triangles; it takes about 1 min to render the image, including generating the knitwear model, depth sorting, and rendering all passes. Most of the time to render the scene is spent on depth sorting, followed by generating the knitwear model. The actual rendering takes only a few seconds.

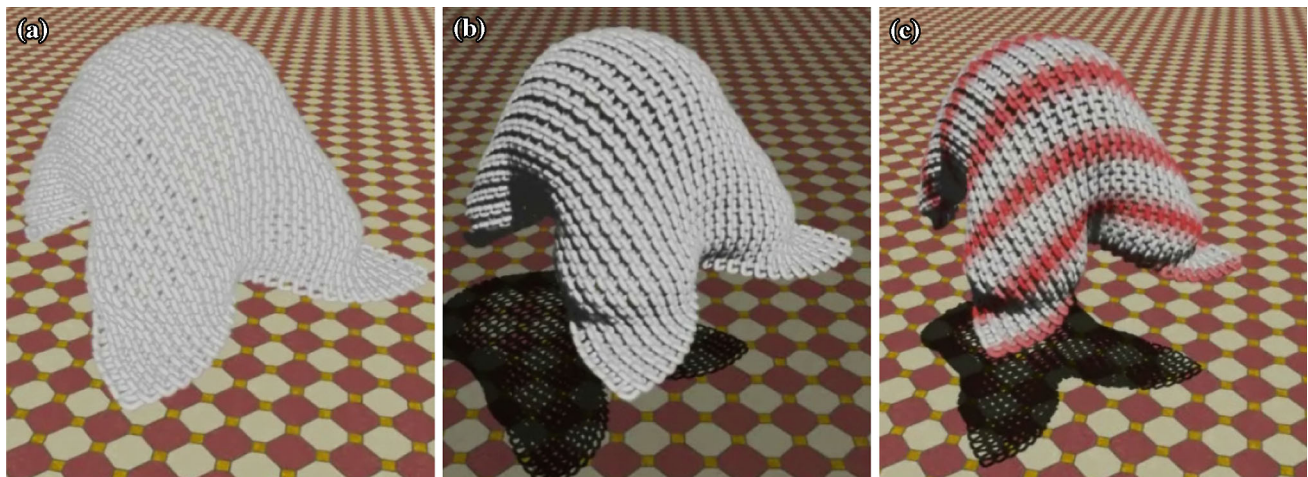


Fig. 6 Knitwear drops onto a rigid object. **a** The scene is rendered with no shadows, **b** with soft shadows, and **c** with soft shadows and texturing. It should be noted that the rigid object is not rendered (see Supplementary material 1)

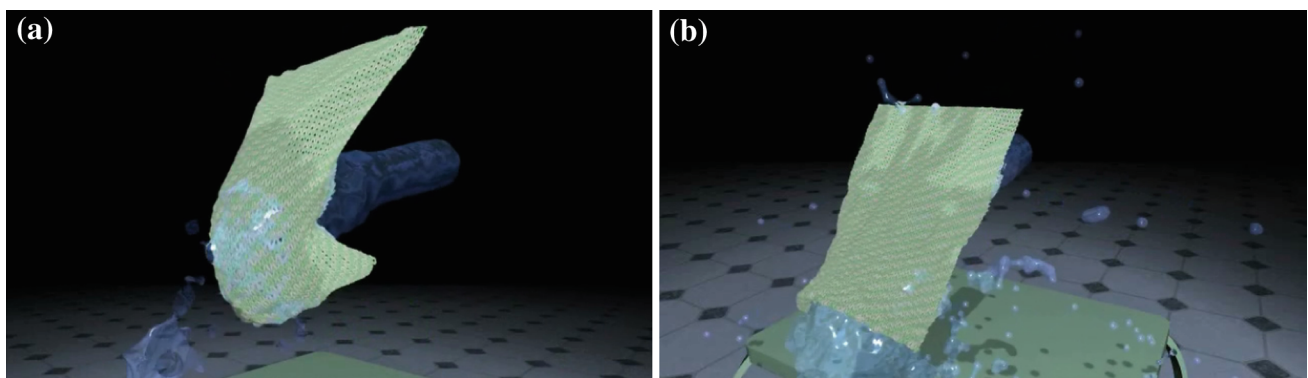


Fig. 7 Still frames from the interaction of fluid with knitwear (see Supplementary material 2)

8 Conclusion

We present a particle-based method to simulate knitwear and its interaction with fluids. The knitwear is simulated by a mass-spring network, and the thread bonding points are interpolated from mass point coordinates. To simulate the volumetric characteristic of knitwear, we use three layers of mass points and connect them by volume-preserving spring constraints. The two-way coupling of knitwear and fluid is implemented by the DEM and capillary pressure-based forces. The fluid is simulated by the SPH.

One shortcoming of the proposed method is that it does not ensure the penetration of the fluid particles to a rigid surface. This, however, does not impede the realism in fluid–knitwear interaction simulation. The surface rendering method we employ produces acceptable results. The main advantage of the simulating and rendering fluid and knitwear using a unified approach is its performance. The challenging task of uniformly rendering a scene consisting of knitwear and fluid is achieved by a hardware-accelerated rasterization-based visualization method. The proposed method provides efficient and practical simulation and rendering of knitwear and its interaction with fluids.

References

- Xu Y., Chen Y., Lin S., Zhong H., Wu E., Guo B., Shum H.: Photo-realistic rendering of knitwear using the Lumislice. In: Proceedings of ACM SIGGRAPH, pp. 391–398 (2001)
- Dimitrov, R.: Cascaded Shadow Maps. http://developer.download.nvidia.com/SDK/10.5/opengl/src/cascaded_shadow_maps/doc/cascaded_shadow_maps.pdf (2011)
- Miller, G.S.P., Pearce, A.: Globular dynamics: a connected particle system for animating viscous fluids. *Comput. Graph.* **13**(3), 305–309 (1989)
- Terzopoulos, D., Platt, J., Fleischer, K.: Heating and melting deformable models. *J. Vis. Comput. Animat.* **2**(2), 68–73 (1991)
- Desbrun, M., Cani, M.-P.: Smoothed particles: a new paradigm for animating highly deformable bodies. In: Proceedings of Eurographics Workshop on Computer Animation and Simulation, pp. 61–76 (1996)
- Monaghan, J.J.: Smoothed particle hydrodynamics. *Annu. Rev. Astron. Astrophys.* **30**, 543–574 (1992)
- Müller M., Charypar D., Gross M.: Particle-based fluid simulation for interactive applications. In: Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 154–159 (2003)
- Solenthaler, B., Schlaffli, J., Pajarola, R.: A unified particle model for fluid–solid interactions. *Comput. Animat. Virtual Worlds* **18**(1), 69–82 (2007)
- Hong, J.-M., Lee, H.-Y., Yoon, J.-C., Kim, C.-H.: Bubbles alive, *ACM Trans. Graph. (Proc. SIGGRAPH)* **27**(3), 4pp (2008) (Article No. 48)
- Losasso, F., Talton, J.O., Kwatra, N., Fedkiw, R.: Two-way coupled SPH and particle level set fluid simulation. In: *IEEE Trans. Vis. Comput. Graph.* **14**(4), 797–804 (2008)
- Ihmsen, M., Akinci, N., Gissler, M., Teschner, M.: Boundary handling and adaptive time-stepping for PCISPH. In: Proceedings of 7th Workshop on Virtual Reality Interaction and Physical Simulation (VRIPHYS), pp. 79–88 (2010)
- Becker, M., Tessendorf, H., Teschner, M.: Direct forcing for Lagrangian rigid-fluid coupling. *ACM Trans. Graph. (Proc. SIGGRAPH)* **15**(3), 493–503 (2009)
- Eberhardt, B., Weber, A., Strasser, W.: A fast, flexible, particle-system model for cloth draping. In: *IEEE Comput. Graph. Appl.* **16**(5), 52–59 (1996)
- Nocent, O., Nourrit, J., Remion, Y.: Towards mechanical level of detail for knitwear simulation. In: Proceedings of WSCG, pp. 252–259 (2001)
- Chen, Y., Lin, S., Zhong, H., Xu, Y., Guo, B., Shum, H.: Realistic rendering and animation of knitwear. In: *IEEE Trans. Vis. Comput. Graph.* **9**(1), 43–55 (2003)
- Kaldor J.M., James D.L., Marschner S.: Simulating knitted cloth at the yarn level. *ACM Trans. Graph. (Proc. SIGGRAPH)* **27**(3), 9pp (2008) (Article No. 65)
- Gröllner, E., Rau, R.T., Straßer, W.: Modeling and visualization of knitwear. In: *IEEE Trans. Vis. Comput. Graph.* **1**(4), 02–310 (1995)
- Gröllner, E., Rau, R.T., Straßer, W.: Modeling textiles as three dimensional textures. In: Proceedings of Eurographics Workshop on Rendering Techniques, pp. 205–214 (1996)
- Durupinar, F., Gudukbay, U.: Procedural visualization of knitwear and woven cloth. *Comput. Graph.* **31**(5), 778–783 (2007)
- Provot, X.: Deformation constraints in a mass-spring model to describe rigid cloth behavior. In: Proceedings of Graphics Interface (GI'95), pp. 147–154 (1995)
- Müller, M., Schirm, S., Teschner, M., Heidelberger, B., Gross, M.: Interaction of fluids with deformable solids. *Comput. Animat. Virtual Worlds* **15**(3–4), 159–171 (2004)
- Cundall, P.A., Strack, O.D.L.: A discrete numerical model for granular assemblies. *Geotechnique* **29**(1), 47–65 (1979)
- Lenaerts, T., Adams, B., Dutré, P.: Porous flow in particle-based fluid simulations. *ACM Trans. Graph. (Proc. SIGGRAPH)* **27**(3), 8 pp (2008) (Article No. 49)
- Chen, Z., Huan, G., Ma, Y.: Computational Methods for Multiphase Flows in Porous Media. Computational Science and Engineering Series. SIAM. <http://www.amazon.com/Computational-Methods-Multiphase-Science-Engineering/dp/0898716063> (2006)
- Leverett, M.C.: Capillary behavior in porous solids. *Am. Inst. Min. Metall. Petrol. Eng.* **142**, 152–169 (1941)
- Bayraktar, S., Gudukbay, U., Özgüç, B.: Particle-based simulation and visualization of fluid flows through porous media. *J. Vis.* **13**(4), 327–336 (2010)
- Bayraktar, S., Gudukbay, U., Özgüç, B.: GPU-based neighbor-search algorithm for particle simulations. *J. Graph. GPU Game Tools* **14**(1), 31–42 (2009)