Technische Universität Dresden • Faculty of Mathematics

Derivation and study of a non-confluent model for deformable cells

Master's thesis

to obtain the second degree

Master of Science
(*M.Sc.*)

written by

Tim Vogel

(born on June 9, 2002 in Finsterwalde)

Day of submission: October 15, 2025

Supervised by Jun.-Prof. Dr. Markus Schmidtchen
(Institute of Scientific Computing)

Contents

Nomenclature

abbreviation	description
PDE	A partial differential equation is an equation that contains unknown multivariable functions and their partial derivatives.
SDE	A stochastic differential equation is a differential equation in which one or more of the terms is a stochastic process, resulting in a solution that is also a stochastic process.
DF model	The discrete form model is a vertex cell model that is defined by a list of all wall points.
N_C	Number of cells in a model
N_V	Number of vertices of each cell in a model

Mathematical conventions

symbol	description
$\vec{v}$	Superscript arrows denote multidimensional variables
$\frac{\partial f}{\partial v}$	Partial derivative of a scalar function f with respect to a one dimensional variable v
$\nabla_{\vec{v}} f$	gradient $\nabla_{\vec{v}} f = (\frac{\partial f}{\partial v_1}, \ldots, \frac{\partial f}{\partial v_n})^T$, where f is a scalar function and $\vec{v} = (v_1, \ldots, v_n)^T$
$D_{\vec{v}} F$	Jacobian matrix $D_{\vec{v}} F = \begin{pmatrix} \frac{\partial F_1}{\partial v_1} \cdots \frac{\partial F_1}{\partial v_n} \\ \vdots \\ \frac{\partial F_m}{\partial v_1} \cdots \frac{\partial F_m}{\partial v_n} \end{pmatrix}$, where $\vec{v} = (v_1, \ldots, v_n)^T$ is a multidimensional variable and $F = (F_1, \ldots, F_m)^T$ is a vector valued function
Δf	Laplacian of a scalar function f
$2e-3$	scientific notation for $2 \times 10^{-3} = 0.002$

Spell check: Use UK spelling (grammarli....)

1 Introduction

Collective cell migration represents a fundamental process underpinning various biological phenomena, including embryonic development, tissue regeneration, wound healing, and the invasive potential of certain cancers. The collective nature of cell movement has been acknowledged for over a century, with early observations recognising its importance in developmental and regenerative processes [AT20, Hol14, Her32, VT66]. However, the underlying mechanisms driving this coordinated behavior remained contentious, with competing hypotheses suggesting roles for pressure [Her32], surface tension [AT20], or active forces generated by leading cells [Hol14]. Following a period where research emphasis shifted towards molecular and genetic details, the field has witnessed a resurgence of interest in understanding the physical principles governing collective cell migration. This revival is largely attributed to recent advances in experimental techniques [RCCT17, DRSB$^+$05, TWA$^+$09], enabling direct measurements of mechanical forces exerted by cells, and the development of new conceptual frameworks in biophysics and active matter physics [MJR$^+$13, PJJ15, JGS18], which challenged purely reductionist perspectives [GT18]. These developments coincided with a growing recognition of the critical role of collective cell migration not only in physiological processes but also in the progression of malignant diseases [FNW$^+$95].

The diverse manifestations of collective cell migration depend heavily on the specific biological context and tissue type [FG09]. For instance, epithelial cells often migrate as cohesive sheets on the extracellular matrix (ECM) during morphogenesis, wound closure, and regeneration. Snapshots from a cell wound healing process, illustrating the dynamics of cell migration, are shown in Figure 1.

In contrast, cancer cells often invade tissues as sheets, strands, or clusters, navigating a complex three-dimensional extracellular matrix (ECM) environment [FG09, CCP$^+$14, CV15].

While cell migration occurs extensively in three dimensions, modeling these complex processes remains a significant challenge. Consequently, much current scientific work focuses on two dimensional systems as a more tractable approach to understand the underlying physical principles. Throughout this thesis, we consider cell dynamics within a bounded two-dimensional domain, denoted by $\Omega \subset \mathbb{R}^2$. Throughout this work, the total number of cells present within the domain Ω is denoted by $N_C \in \mathbb{N}$.

In two dimensions, cell monolayers serve as fundamental model systems to study cell behavior and tissue function. These systems, comprising a single layer of cells grown on a surface, can be mathematically and computationally modeled in either a confluent or non-confluent manner. A confluent cell model depicts a continuous, tightly packed layer where cells cover the entire surface without gaps, whereas a non-confluent monolayer represents a state with spaces and gaps between individual cells or cell clusters that have not yet achieved full surface coverage.

Recent years have seen growing interest in understanding the principles governing collective cell migration in confluent cell monolayers and epithelial tissues, which exhibit remarkable patterns and correlations in both structural arrangements and actively driven flows [WV21]. Experimental studies on model systems have revealed phenomena such as unjamming transitions, spontaneous vortex formation, topo-

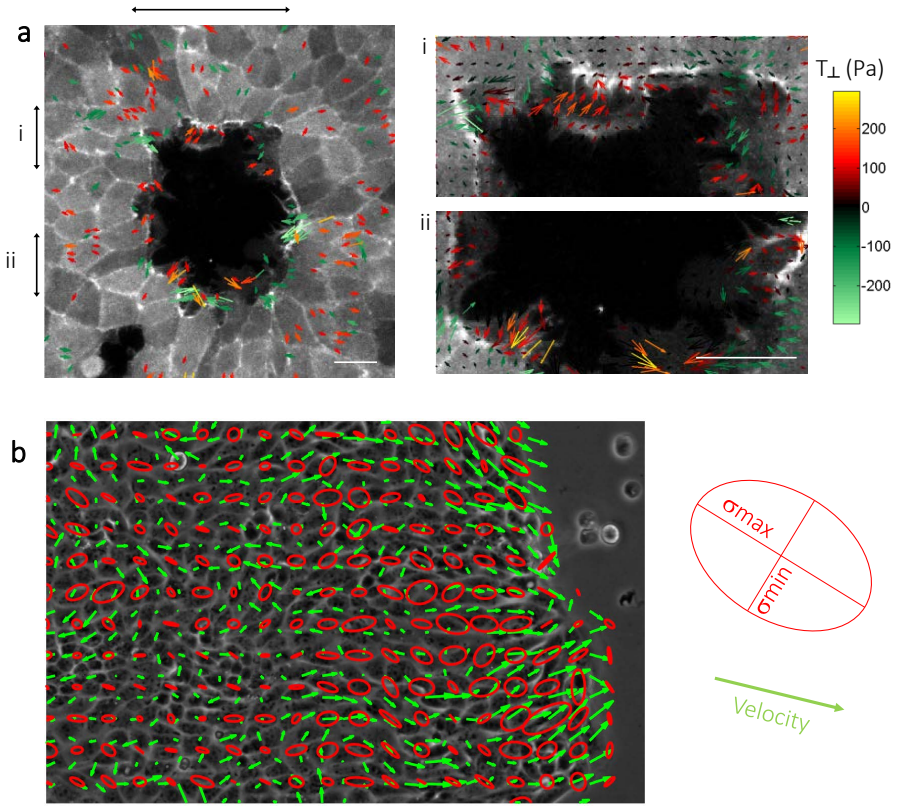

Figure 1: A figure from the paper [AT20] illustrating cell migration during the wound healing process. (a) Vector field representation of cellular traction forces in cells as they close a wound, with color intensity reflecting the radial component (positive values indicate outward-directed forces). Panels labeled i and ii present magnified views of the regions marked by arrows in panel (a). (b) Velocity vectors (green) and monolayer stress ellipses (red), depicting the principal stress directions and magnitudes, in a growing cell colony (phase contrast).

logical defects, and active turbulence. A key challenge is linking this macroscopic behavior to the properties of individual cells and their interactions, leading to a diverse range of modeling approaches spanning different levels of coarse-graining, from subcellular lattices and multiphase field models to vertex, Voronoi, particle, and continuum models. The systematic comparison of these diverse cell models is crucial for selecting appropriate methods for future studies and enabling predictive simulations of patterns and correlations in cell colonies.
Understanding the diffusion behavior of cells, influenced by various forces and interactions, is a key aspect of collective cell migration. This will be a central focus of this thesis. Different mathematical cell models, incorporating distinct cell dynamics, will exhibit varying diffusion behaviors. To begin this investigation, we will first introduce the simplest model: the point particle model.

Point particle model
We consider the point particle model on a two dimensional bounded domain $\Omega \subset \mathbb{R}^2$, where we have $N_C \in \mathbb{N}$ particles. These particles have no real size. There is also no

particle interaction, as there is no possibility of collision.
Initially, the particles are randomly distributed in Ω.
The particles' dynamics are governed solely by Brownian motion. Brownian motion is a random and unpredictable motion that occurs in the real world when particles are suspended in a fluid and collide with surrounding molecules.
In mathematics, we model Brownian motion using stochastic differential equations (SDEs), which are equations that describe the motion of a particle over time in a random and unpredictable manner. SDEs are a powerful tool for modeling complex phenomena in physics, finance, and other fields, and are characterized by the presence of random terms that capture the uncertainty of the system.
Let

$$\vec{x}_i(t) \in \Omega \quad 1 \leqslant i \leqslant N,$$

be the location of the particle i at time $t > 0$. The particle movement can be modeled using the diffusion equation, which describes the random motion of particles over time

add f term for chemical gradient or environmental impact

$$d\vec{x}_i(t) = \sqrt{2D}\, dB_t^{(i)}, \quad 1 \leqslant i \leqslant N,$$

where the constant $D > 0$ represents the diffusion coefficient which proportionally scales the speed of the particle movements by scaling the random fluctuations. The term $dB_t^{(i)}$ introduces the randomness of Brownian motion, where $dB_t^{(i)}$ is a normally distributed random variable that accounts for the unpredictable changes in the position of particle $vecx_i$ over time.
We also consider the probability density function $\rho(t, \vec{x})$, which describes the probability of finding a particle at a specific position $\vec{x}$ at time t. In the given context, the function ρ satisfies the partial differential equation:

$$\frac{\partial \rho(t, \vec{x})}{\partial t} = D\Delta_{\vec{x}}\rho(t, \vec{x}), \tag{1}$$

where $\Delta_{\vec{x}}$ is the Laplacian operator with respect to the spatial variables.
Equation (1) represents the classic diffusion equation, a cornerstone of physics and mathematics. The same diffusion constant $D > 0$ is used in the SDE for particle movement and the PDE for the probability density function ρ.

Hard sphere model

Next, we consider models that add a real size to the particles and introduce particle interactions. With the inclusion of a real size, the particles cannot overlap, resulting in exclusion effects. To account for this, we introduce a new interaction dynamics that ensures the particles do not overlap. This new interaction dynamics leads to a more complex and realistic model that captures the behavior of particles with a real size and interactions.
Since particles cannot overlap, the domain $\Omega_\epsilon^{(i)}$, that holds the information where the centre of particle i can be located, must exclude the areas where $\|\vec{x}_i - \vec{x}_j\|_2 \leqslant \epsilon$ for all $1 \leqslant j \leqslant N_C$, $j \neq i$. This is due to the fact that particles cannot occupy the same space simultaneously.
The domain that holds all possible locations of the particles is then given by

$$\Omega_\epsilon^{N_C} = \Omega_\epsilon^{(1)} \times \ldots \times \Omega_\epsilon^{(N_C)}.$$

look at paper p. 2 down left

This can be visualized as a product space, where each particle's domain is combined to form a larger domain that encompasses all possible locations of the particles. Under this circumstances, we will get a new dynamic compared to the point particle model. Figure 2 illustrates how a hard sphere cell configuration looks like compared to a point particle configuration.

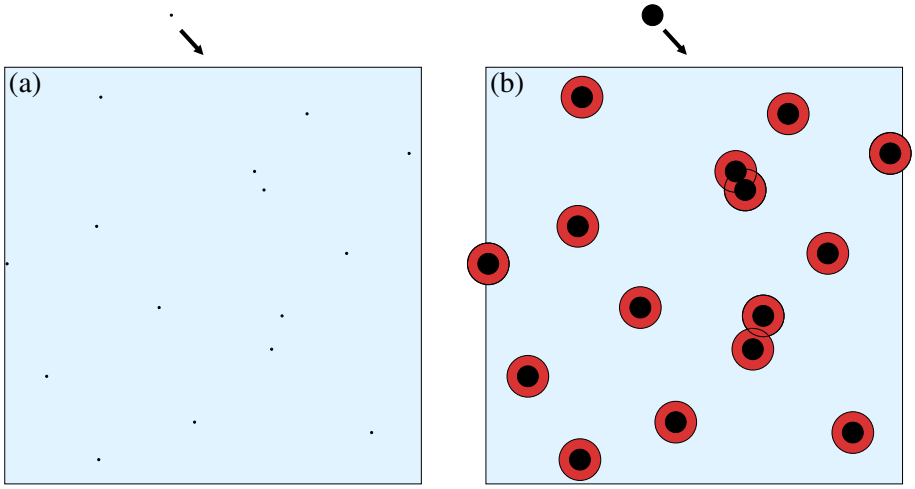

Figure 2: An illustration from [BC12] of the point particles on the left and the hard sphere particles on the right. We can see areas where the centre of the particles cannot be located, as they would overlap with other particles being marked red.

In the work of Bruna et al. [BC12], a hard sphere particle model is examined. Here, the particles are spherical in shape, with a diameter $0 < \epsilon \ll 1$. All particles in this model are distinct and can be distinguished from one another. The diffusivity constant D is set to be $D = 1$.
The hard sphere model is characterized by the fact that any interaction between particles may cause a change in their direction of motion, but the spherical shape of the particles remains unchanged. Hardcore collisions are modeled as reflective boundary conditions on the collision surfaces defined by $r = \|\vec{x}_i - \vec{x}_j\|_2 = \epsilon$, where $1 \leqslant i < j \leqslant N_C$. The external forces acting on a particle in the system are described by the force function $f : \mathbb{R}^2 \to \mathbb{R}^2$, which depends only on the location of the particle. The function $\vec{F}$ maps the particle configuration $\vec{X} = (\vec{x}_1, \ldots, \vec{x}_{N_C})^T$ to the vector of external forces $\vec{F}(\vec{X}) = (f(\vec{x}_1), \ldots, f(\vec{x}_{N_C}))^T$. The dynamics of the particles are governed by the SDE

$$d\vec{x}_i(t) = \sqrt{2D}\, dB_t^{(i)} + f(\vec{x}_i(t))\, dt, \qquad 1 \leqslant i \leqslant N_C.$$

In this model, the particles are initially randomly distributed in $\Omega_\epsilon^{N_C}$, ensuring that no overlap occurs between the particles. The joint probability density function P of the N_C particles satisfies the high-dimensional Fokker-Planck equation

$$\frac{\partial P}{\partial t} = \nabla_{\vec{X}} \cdot (\nabla_{\vec{X}} P - P\vec{F}),$$

where $\nabla_{\vec{X}}$ and $\nabla_{\vec{X}}\cdot$ denote the gradient and divergence operators with respect to the N_C-particle position vector $\vec{X}$.

Using the method of matched asymptotic expansions, the authors also derived the probability density function ρ of finding a single particle at time t and position $\vec{x}$, which satisfies the equation

$$\frac{\partial \rho(t,\vec{x})}{\partial t} = \Delta_{\vec{x}}\rho + \frac{\pi}{2}(N_C - 1)\epsilon^2 \Delta_{\vec{x}}(\rho^2) - \nabla_{\vec{x}} \cdot (f(\vec{x})\rho). \tag{2}$$

When f is neglected and $\epsilon \to 0$, this equation reduces to the probability density function of the point particle model, except for a rescaling factor. The Fokker-Planck equation is a direct extension of the diffusion equation, with an additional drift term.

A central finding of [BC12] is shown in Figure 3, which compares the diffusion behavior of the point particle model and the hard sphere model. It uses the Monte Carlo method for the point particles and the hard sphere in order to give a numerical approximation of the density dynamic. The Monte Carlo method involves running the simulation many times ($M = 10000$). For each discrete subsquare of our domain, we count how many particles were found at that location across all runs at time $t = 0.05$. We then divide by the total number of runs to get the estimated probability density at that place and time. This estimated density converges to the true density as the number of runs increases.

Soft sphere model

Next, we consider an extension of the model by introducing deformable soft spherical particles. This new model incorporates the effect of deformation and interaction between particles through a potential energy function that depends on the distance between the particles. The paper [BCR17], written by Bruna, Chapman and Robinson, analyses the diffusion properties of such a model.

In contrast to the soft sphere model, the hard sphere model enforces rigid, non-deformable cell boundaries through reflective boundary conditions at a fixed separation distance, leading to abrupt, instantaneous collisions without any interface deformation. The soft sphere model uses a smoother approach including interaction potentials.

The equation of motion for each particle i is given by

$$d\vec{x}_i(t) = \sqrt{2D}\, dB_t^{(i)} + f(\vec{x}_i(t))\, dt - \sum_{j \neq i} \nabla_{\vec{x}_i} u(\|\vec{x}_i(t) - \vec{x}_j(t)\|_2)\, dt, \qquad 1 \leqslant i \leqslant N_C,$$

where $\nabla_{\vec{x}_i}$ is the gradient with respect to $\vec{x}_i$.

The effect of the interaction potential is to cause particles to repel or attract each other depending on the distance between them, rather than simply overlapping.

For the modeling of short range interacting soft sphere particles, the authors computed the one particle probability density $\rho(t,\vec{x})$ of finding a given particle at position $\vec{x}$ at time t developing according to

$$\frac{\partial \rho}{\partial t} = \nabla_{\vec{x}} \cdot (D\nabla_{\vec{x}}\rho - f(\vec{x})\rho + \alpha_u \epsilon_u^2 (N_C - 1)\rho \nabla_{\vec{x}}\rho) \tag{3}$$

where α_u depends on the interaction potential u and $0 < \epsilon_u \ll 1$ is the interaction range of u. When comparing the first marginals of the soft sphere model Equation (3) and the hard sphere model Equation (2), we can see that they are similar in

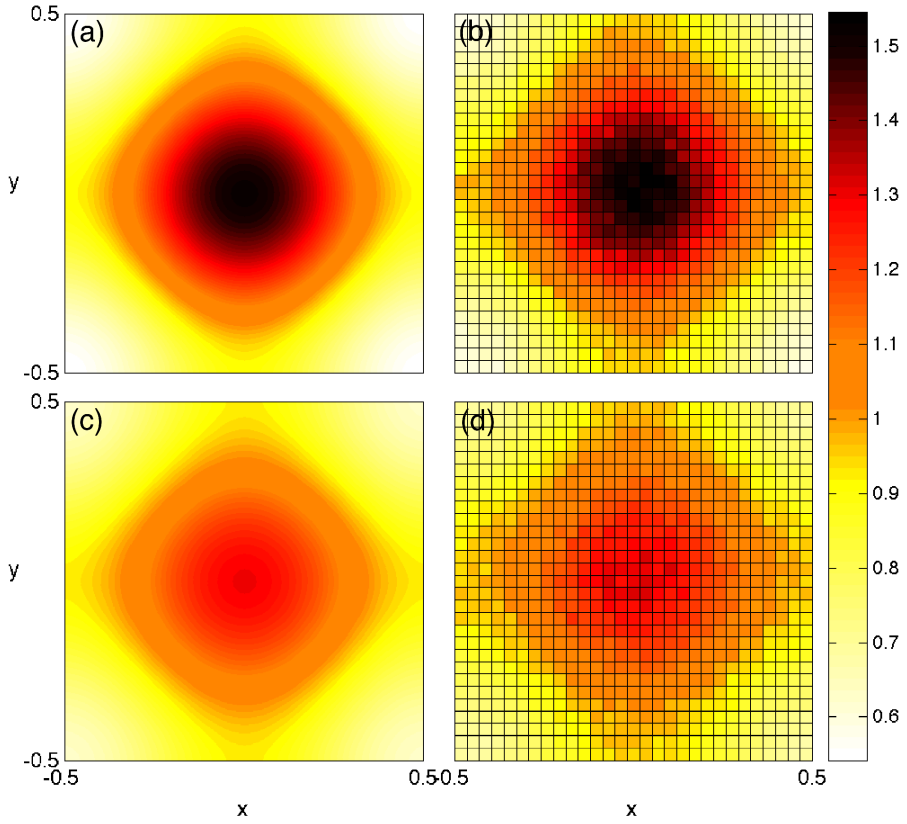

Figure 3: This figure from [BC12] contains the following four plots, all of them are shown at time $t = 0.05$. For all plots, the initial condition is normally distributed with mean $(0, 0)^T$ and standard deviation 0.09. (a) shows the solution of the linear diffusion Equation (1) for point particles. (b) shows the histogram of a Monte Carlo simulation of the point particle model. (c) shows the solution of the nonlinear diffusion Equation (2) for finite-sized particles. (d) shows the histogram of a Monte Carlo simulation of the hard sphere model. The Monte Carlo simulations used 10^4 simulation runs each with a time step size of 10^-5. We can see that the hard sphere model in (c) and (d) shows a quicker diffusion rate as the cell concentration in the centrum of the domain has already diffused more compared to the point particle model in (a) and (b).

structure, with the main difference being the coefficients of the nonlinear diffusion terms. We can not clearly say which model diffuses faster, as this is dependent on the modelling of the soft interaction.
The influence of the cell hardness to the diffusion rate of the cell system will be investigated in this thesis. We even introduce a parameter that can continuously change the cell hardness from hard to soft.
While these models are powerful, they are limited to spherical particles and do not account for the complex shapes and deformations observed in biological cells.

Phase field model
A new cell modeling framework is now considered. The phase field approach exhibits

conceptual parallels with the soft sphere model proposed by Bruna, Chapman, and Robinson [BCR17], particularly in how cell-cell interactions are modeled through a continuous repulsive energy that prevents overlap.
In both formulations, interactions are governed by a smooth, short range influence. The soft-sphere model derives its interaction dynamics from the potential energy $u(\|\vec{x}_i - \vec{x}_j\|_2)$, which leads to a nonlinear diffusion equation featuring a diffusion coefficient that depends on local density.
The phase field model resolves cell overlaps through a free energy producing a repulsive force between cells that scales with the local concentration of ϕ_i. This enables gradual, continuous deformation of cell interfaces, thereby avoiding discontinuities in the dynamics that are characteristic of discrete collision models. Thus, the interaction mechanisms operate continuously and locally, ensuring a seamless transition between overlapping and non overlapping configurations while maintaining physical consistency in both models.
Nevertheless, the phase field model diverges fundamentally in its underlying structure: it is a continuum model based on partial differential equations that explicitly encodes cell morphology and internal organisation through the phase field variable ϕ_i, enabling dynamic shape changes, topological transitions, and coupling to geometric features such as surface curvature.
The free energy functional encodes shape regularization, intercellular interactions, and physical constraints of the system. Unlike the soft-sphere model, which treats cells as point-like entities interacting via a smooth potential, the phase field model represents cells as much more complex, spatially extended, continuously structured entities, with their internal state fully described by the evolution of the phase field variable ϕ_i. This enables a natural description of complex cell morphologies, topological transitions such as cell division or fusion, and the integration of cell mechanics with geometric curvature via extrinsic curvature contributions in the free energy.
Phase field variables represent cells as smooth functions $\phi_i(\vec{x}, t) \in [-1, 1]$, with $\phi_i > 0$ in the cell interior and $\phi_i < 0$ in the exterior. The cell wall is denoted by values of $\phi_i = 0$. An illustration of a phase field variable is shown in Figure 4.

The dynamics of a phase field variable ϕ_i can typically be given by a gradient flow of a free energy functional:

$$\frac{\partial \phi_i}{\partial t} + v_0(\vec{v}_i \cdot \nabla_{\vec{x}} \phi_i) = \Delta_{\vec{x}} \frac{\delta F}{\delta \phi_i}, \qquad 1 \leqslant i \leqslant N_C$$

where $\vec{v}_i$ is a vector field used to incorporate activity, with a self propulsion strength v_0, F is a free energy, and $\dfrac{\delta F}{\delta \phi_i}$ denotes the first variation.
The free energy F arises from a sum of different energies,

$$F = F_{CH} + F_{INT} + F_M.$$

The first energy is a Cahn-Hilliard energy and could look like in [WV21]

$$F_{CH} = \sum_{i=1}^{N} \int_\Omega \frac{1}{Ca} \left(\frac{\epsilon}{2} \|\nabla_{\vec{x}} \phi_i\|_2^2 + \frac{1}{\epsilon} W(\phi_i) \right) d\vec{x},$$

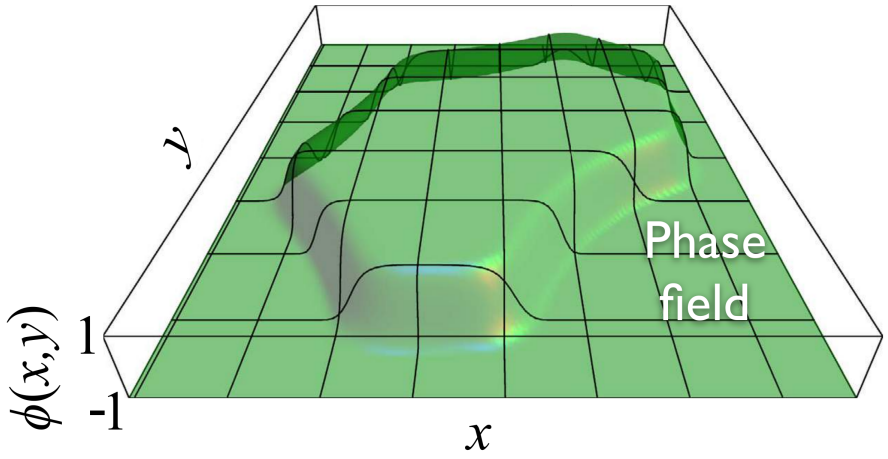

Figure 4: A snapshot from the paper [AT20] illustrating a phase field variable. The cell's inside has value $\phi = 1$ and the outside $\phi = -1$.

where ϵ is a small parameter related to the interface thickness, and Ca is a capillary number that scales the relative importance of surface tension. The term $\|\nabla_{\vec{x}}\phi_i\|_2^2$ penalises a long cell wall, as $\nabla_{\vec{x}}\phi_i \neq 0$ only at the cell wall. Thus, the Cahn-Hilliard energy always tries to minimize the area of the cell interface.
$W(\phi_i) = \frac{1}{4}(\phi_i^2 - 1)^2$ is a double-well potential. This energy ensures that each ϕ_i maintains a stable interface of $[-1, 1]$.
The second energy term F_{INT} models cell-cell interactions and could be defined as in [WV21]

$$F_{INT} = \sum_{i=1}^{N} \frac{1}{Ca} \int_{\Omega} B(\phi_i) \sum_{j \neq i} w(d_j)\, d\vec{x},$$

where

$$B(\phi_i) = \frac{3}{4\sqrt{2}\epsilon}(\phi_i^2 - 1)^2$$

is an approximation of the delta function of the cell boundary that is non-zero only at the cell wall. The sum in the integral accounts for the interaction with all other cells $j \neq i$ through a short-range potential $w(d_j)$, where d_j is the signed distance function to the cell boundary of cell j. When employing this dynamics, the interaction force F_{INT}, acts to reduce or eliminate cell overlaps.
The third energy term F_M differs for different models and incorporates additional mechanical properties of the cells, such as area conservation or bending energy. This energy is dependent on the use case of the model.
It can have a big influence on the dynamic of a phase field model, as analysed in [WV21], where the authors focussed on the influence of microscopic details to incorporate active forces on emerging phenomena.
Four different approaches are considered. One in which the activity is determined

by a random orientation, one where the activity is related to the deformation of the cells, and two models with subcellular details to resolve the mechanochemical interactions underlying cell migration.
The random model determines the direction of motion on the single cell level by a stochastic process. The second model is called elongation model. It identifies the longest axis of the cell's phase field and aligns the direction of motion with this axis. The third and fourth models presented are referred to as the polar model and the nematic model, respectively. Cell motion strength and direction are determined based on subcellular details at the single-cell level.
All models are compared with respect to generic features, such as coordination number distribution, cell shape variability, emerging nematic properties, as well as vorticity correlations and flow patterns in large confluent monolayers and confinements. We can see the results in Table 1.

The goal of this paper is a systematic comparison of these approaches and their linkage with statistical observables of experiments to provide a route towards predictive simulations of patterns and correlations in cell colonies. Model predictions are compared with experimental data from various cell cultures. The qualitative differences observed highlight the importance of microscopic details.
A related study by Happel and Voigt [HV23] employs phase field models to investigate cell dynamics, specifically focusing on the influence of curved domains (defined as tori) on collective cell behavior. The visualization of cells on a torus is shown in Figure 5.
Their work examines emergent phenomena like coordinated rotation on curved surfaces, driven by curvature alignment and self-propulsion. Unlike the soft-sphere model, which typically assumes spherical symmetry and isotropic interactions, the phase field approach allows for the incorporation of anisotropic effects, such as alignment with principal curvature directions, through geometric coupling terms. This capability enables the simulation of complex collective behaviors but also increases computational demands compared to simpler point-particle models.
The phase field model developed by Happel and Voigt [HV23] highlights the critical role of extrinsic curvature coupling in dictating the alignment of cell elongation with principal curvature directions and the emergence of coordinated rotation in epithelial layers on curved surfaces. By combining a diffuse interface representation with a free energy incorporating both intrinsic and extrinsic geometric terms, their simulations successfully reproduce key experimental observations, including spontaneous rotation on cylindrical surfaces and curvature-dependent morphological changes on tori. This work emphasizes the significance of geometric constraints in tissue morphogenesis and offers a framework for investigating how curvature influences collective cell dynamics beyond planar environments.

Vertex model
The last cell model approach is also the approach we choose in this thesis. Vertex models represent a powerful and versatile approach for simulating the mechanical behavior of biological cells and tissues.
Originating from materials science and solid mechanics, these models represent cells

characteristic	Random	Elongation	Polar	Nematic
Coordination number distribution	(✓)	(✓)	(✓)	(✓)
Shape variability	(✓)	✓	✓	(✓)
Rosette ratio		Differences between models		
Velocity distribution of topological defects		Differences between models		
Correlation between direction of motion and orientation of defect	×	✓	✓	(✓)
Elastic property of + 12 defect	×	Extensile	Contractile	Contractile
Active turbulence	(✓)	(✓)	(✓)	(✓)
Vorticity-vorticity correlation		Similar for all models		
Dependency of defect density on activity	Linear	Linear	Linear	Constant
Rotational motion in circular confinement	×	(✓)	×	×

Table 1: Comparison of the four different phase field models from [WV21] with respect to various characteristics observed in experiments. A check mark ✓ indicates observed agreement, × indicates disagreement and (✓) indicates only qualitative agreement with universal feature. If experimental data are not available or insufficient for a comparison, only similarities or differences of the models are noted.

as polygons, where the boundaries are defined by a discrete set of vertices connected by edges. The degrees of freedom in these models are precisely these vertices, meaning that all the cell model dynamics are applied only to the vertices.
The cell dynamic in a vertex model is usually given by the equation

$$\frac{d\vec{x}_i}{dt} = F_i^m, , \tag{4}$$

where F_i^m is the total force acting on vertex $1 \leqslant i \leqslant N_V$ of cell $1 \leqslant m \leqslant N_C$.

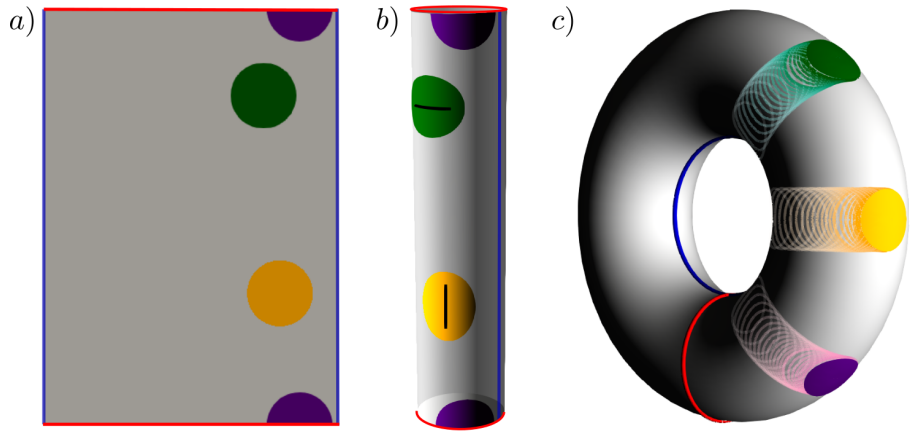

Figure 5: A figure from the paper [HV23] showing a phase field variable ϕ on a curved domain, specifically a torus. In (a) and (b), the surface of the torus from (c) is shown as a parametrization on a rectangle, where the blue edges are identified (glued together) to form the toroidal direction, and the red edges are similarly identified to form the poloidal direction. Red and blue lines represent periodic boundary conditions, which are enforced by gluing the corresponding edges in (b) and (c). The color coding corresponds to the extrinsic curvature parameter E_c (see Eq. (3)): $E_c = 0$ (purple) results in a geodesic circle on both geometries; $E_c > 0$ (green) favors alignment with the direction of maximum absolute curvature; $E_c < 0$ (yellow) favors alignment with the direction of minimal absolute curvature. Cell elongation is highlighted for visibility. On toroidal surfaces, cell shape depends on position due to varying curvature. (c) shows the trajectories, final positions, and shapes of cells over time. The influence of extrinsic curvature is not apparent in the final configuration, as all shapes were obtained by solving Eq. (1) with $v_0 = 0$.

Like in the phase field model, F_i^m is a sum of different forces that define the cell behavior, such as the cell flexibility or the interaction with other cells. The forces participating in F_i^m or often found as negative gradients of energies that shall be minimized. This method called gradient flow is also used for phase field models.
Due to their discrete nature, vertex models are computationally efficient compared to continuum models, while still providing rich insights into the interplay between cell mechanics and collective behavior, making them invaluable for understanding fundamental principles governing cell organization and tissue dynamics across diverse biological contexts.
This geometric representation allows for the incorporation of key cellular properties, such as surface tension, cortical tension, adhesion, and local shape constraints, into an energy functional. Minimizing this energy through computational simulation enables the capture of emergent phenomena like tissue morphogenesis, cell migration, wound healing, and pattern formation.

The paper [FOBS14] developed a sophisticated vertex based model to simulate the dynamic behavior of confluent epithelial cell sheets, representing tissues where cells completely cover the available space without gaps. Their model aims to capture key biological processes observed in real epithelial tissues.

Conservation of cell area and perimeter ensures cells maintain their size and shape during deformation. Junctional rearrangements, including neighbor exchange and vertex/edge merging, enable cell migration and tissue remodeling. Cell division is modeled by creating a rosette structure from multiple adjacent vertices and cell growth and death simulate changes in individual cell size. A detailed exploration of the implementation and specific consequences of these confluent specific mechanics is beyond the scope of this work. While these mechanisms are essential for understanding the complex dynamics of confluent tissues, the focus of this thesis lies on non confluent cell models, where the presence of interstitial space and variable packing density leads to distinct physical behaviors.

Boromand, Merkel, and Manning [BSY+18] investigated the jamming transition in a system of deformable cells using a vertex model. Of particular interest is their development of a non confluent cell model meaning that gaps between the cells are allowed by the model. This model is versatile and can be applied to simulate cells, foams, emulsions, and other soft particulate materials.
It uniquely combines the ability to represent individual deformable particles with the established shape energy function of the vertex model. This shape energy function incorporates terms that penalize deviations from a target area (a) and perimeter (p), along with repulsive interparticle forces to prevent overlap.
The model defines polygons with $N_V \in \mathbb{N}$ vertices, where the bond vector $\vec{l}_{mi}$ connects vertex $\vec{v}_{m,i}$ to $\vec{v}_{m,i+1}$, i.e. vertex i to vertex $i+1$ of cell m. The complete shape energy function, which governs the particle dynamics, integrates these various contributions:

$$U = U_{\text{contract}} + U_{\text{compress}} + U_{\text{line tension}} + U_{\text{bending}} + U_{\text{interaction}}.$$

The first energy reads

$$U_{\text{contract}} = \frac{k_l N_V}{2} \sum_{m=1}^{N_C} \sum_{i=1}^{N_V} (l_{m,i} - l_0)^2,$$

where k_l is the spring constant and l_0 is the equilibrium length of the edges. It penalises contractions or expansion of cell edges and makes the edges return to their desired length l_0.
Similarly, the second force U_{compress} stabilises the cells area to a equilibrium area a_0

$$U_{\text{compress}} = \frac{k_a}{2} \sum_{m=1}^{N_C} (a_m - a_0)^2,$$

where k_a is compressibility constant.
The line tension energy is another energy working with the cell edges. It sanctions long edges with the formula

$$U_{\text{line tension}} = \gamma \sum_{m=1}^{N_C} \sum_{i=1}^{N_V} l_{m,i},$$

where γ is the line tension coefficient.
The last shape energy is given by the bending energy. It acts to resist changes in the angles between adjacent edges of the polygonal cells. This energy encourages the cells to maintain smooth, relatively straight boundaries and discourages sharp bends or kinks.

$$U_{\text{bending}} = \frac{k_b}{2N_V} \sum_{m=1}^{N_C} \sum_{i=1}^{N_V} \left(\frac{2(\hat{l}_{m,i} - \hat{l}_{m,i+1})}{l_{m,i} - l_{m,i+1}} \right)^2,$$

where k_b is the bending rigidity constant, $\hat{l}_{m,i} = \frac{\vec{l}_{m,i}}{\|\vec{l}_{m,i}\|_2}$ is the unit vector of $\vec{l}_{m,i}$. These four energies are conserving a specific cell shape.

There are two different methods to model the repulsive interaction between two

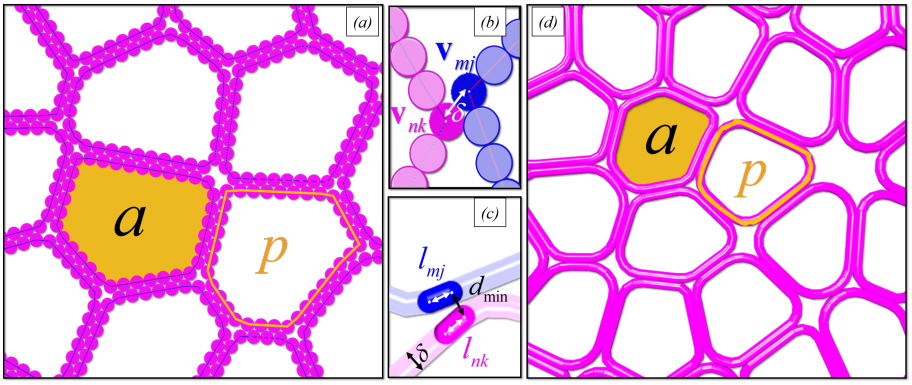

Figure 6: A snapshot from the paper [BSY+18] illustrating a configuration of vertex-based deformable cells. Schematic of deformable polygons with $N_V = 34$ vertices (where the position of the j-th vertex in the m-th polygon is denoted by $\vec{v}_{m,j}$), area a_m, and perimeter p_m. The edge $l_{m,j} = p_m/N_V$ represents the line segment connecting vertices j and $j+1$ in polygon m. Two methods are used to model the edges of deformable polygons: (a) and (b) show the RS method, where disks of diameter δ are centered at the polygon vertices; (c) and (d) show the SS method, where polygon edges are modeled as circulo-lines of width δ. The quantity $d_{\min}$ denotes the minimum distance between the line segments $l_{m,j}$ and $l_{n,k}$.

deformable polygons called rough surface (RS) and smooth surface (SS) method. Both methods are illustrated in Figure 6.
In the Rough Surface (RS) method, the discrete nature of the polygonal cell representation is emphasized. Each vertex of a cell is treated as the center of a rigid disk with a fixed diameter, set to $\delta = l_0$ (equilibrium bond length). Repulsive interactions between cells are then modeled as linear spring forces that arise when these disks overlap. This approach effectively simulates a rough surface interaction, characterized by discrete points of repulsion localized at the cell vertices, rather than a continuous interaction across the cell boundary.

$$U_{\text{RS}} = \sum_{m=1}^{N_C} \sum_{n>m}^{N} \sum_{j=1}^{N_V} \sum_{k=1}^{N_V} \frac{k_r}{2} (\delta - |\vec{v}_{m,j} - \vec{v}_{n,k}|)^2 \times \Theta(\delta - |\vec{v}_{m,j} - \vec{v}_{n,k}),$$

where k_r is the repulsive constant and Θ is the Heaviside step function, that is either 1 for a positive argument or 0 otherwise.
In contrast to the RS method, the Smooth Surface (SS) method adopts a smoother representation, modeling polygon edges as circulo-lines, essentially line segments with a finite width δ. Repulsive forces are then calculated based on the minimum distance $d_{\min}$, between these circulo-line segments belonging to different polygons. The interaction energy is then formulated using this minimum distance $d_{\min}$.

$$U_{\mathrm{SS}} = \sum_{m=1}^{N_C} \sum_{n \neq m} \sum_{j=1}^{N_V} \sum_{k=1}^{N_V} \frac{1}{2} k_r \left(\delta - d_{\min}\right)^2 \Theta\left(\delta - d_{\min}\right).$$

This method provides a smoother, more continuous repulsion, better approximating the behavior of soft, continuous interfaces.
Despite the different interaction mechanisms, both methods yield similar structural and mechanical properties at jamming onset. This indicates that the overall jamming behavior is robust to the specific choice of interaction model.

Discrete Form model
This thesis introduces the Discrete Form (DF) cell model, a vertex based framework inspired by [FOBS14, BSY$^+$18] for simulating cell dynamics in non-confluent systems, similar to [BSY$^+$18], but typically using fewer vertices (e.g., $N_V = 6$).
The DF model incorporates shape preserving forces (area, edge, interior angle energies) and interaction forces (deforming and bounce overlap forces) derived as gradient flows of energy functionals, allowing for cell deformation like in [BSY$^+$18, FOBS14, WV21, HV23] and ensuring evolution towards lower-energy configurations. This allows representation of a wide range of desired cell shapes.
The core contribution of this thesis is a non confluent DF model that systematically investigates how cellular deformability, controlled by a hardness parameter $h \in [0, 1]$, influences the overall diffusivity of the cell system, enabling a transition from hard disc-like to soft disc-like behavior. Building upon the foundations laid in the Bachelor's thesis [Vog23], significant improvements were made, including bug fixes, code parallelization, and reformulation of forces for stable large-scale simulations (detailed in Sections 3 and 4).
Chapter 5 presents a rigorous validation by recreating the diffusion dynamics of the point particle and hard sphere models from [BC12] through extensive Monte Carlo simulations with varying hardness ($h \in \{0, 0.5, 1\}$), representing soft, intermediate, and hard cell behaviors.
Subsequently, Chapter 6 analyzes the resulting density dynamics. By connecting to the frameworks of [BC12, BCR17, BSY$^+$18], this work provides a unified perspective on cell dynamics spanning rigid to deformable regimes. Finally, the thesis concludes with a summary of findings and an outlook on potential future extensions and applications of the DF model.

2 TODOs

Cell dynamics - SDE

- add SDE for vertices

Sanity Check

- write down the results
- analyse them -¿ is there much to say? we only see a faster diffusion for harder cells
- start a new sim with needles? stability issues, how solve them?
- add those aquator diagrams

Density comps

- think about density pde. Remember edge computation with edge energy and 2 vertices cells (photo!)
- read robins BT, learn how this works
- the shape preserving forces dont change density
- What to do with overlap forces?
- write everything down

general things

- add left(, right) where it is necessary
- control indices
- check -
- add references to introduction papers to all chapters
- be consistent in naming of DF model

3 DF cell model

The following two sections are a recap of the DF cell model and its dynamics that were introduced in my Bachelor's thesis [Vog23].
We are considering cells in the two dimensional space $\mathbb{R}^2$. Here, cells are considered to be polygons.

Definition 3.1. Polygon
A polygon is a closed geometric figure in $\mathbb{R}^2$, constructed by joining a finite number of straight line segments end to end. It can be described by a sequence of its vertices $(\vec{v}_1, \dots, \vec{v}_N)$. The following properties characterise a polygon:

1. A polygon is **simple** if no two line segments cross each other.
2. A polygon has a **positive orientation** if the vertices are ordered counterclockwise.
3. A polygon has a **negative orientation** if the vertices are ordered clockwise.

Having established this definition, we are now ready to define our cell model.

Definition 3.2. Discrete form (DF)
A cell in its discrete form (**DF**) is given by an ordered sequence of its vertices $C = (\vec{v}_1, \dots, \vec{v}_N)$ if the resulting polygon when connecting every vertex with its neighbours and $\vec{v}_1$ with $\vec{v}_N$ is simple and positively orientated. We set $\vec{v}_{N+1} = \vec{v}_1$ and $\vec{v}_0 = \vec{v}_N$ to enable periodic indexing, which simplifies the computation of the upcoming forces a lot.

In this thesis, DF cells may also be called discrete cells. In our model, the cell vertices are denoted by $\vec{v}$. Thus, the character v refers to vertex positions and not to velocity. The term velocity is not used throughout this thesis as vertex dynamics are entirely given by the upcoming forces and a cell wise computed Brownian motion.

The next step is to describe the setup of a DF simulation.

Definition 3.3. DF simulation
A DF simulation considers $N_C \in \mathbb{N}$ cells. Each cell has the same amount of $N_V \in \mathbb{N}$ vertices. Thus, the notation of all cells and their vertices is given by

$$C^i = (\vec{v}_1^{\,i}, \dots, \vec{v}_{N_V}^{\,i}), \quad 1 \leqslant i \leqslant N_C.$$

The complete set of all cells is represented by

$$\vec{C} = (C^1, \dots, C^{N_C}),$$

which also contains all vertices from all cells.
The simulation's dynamics are defined on all cell vertices via the stochastic differential equation (SDE):

$$\mathrm{d}\vec{v}_j^{\,i}(\vec{C}) = \mathbf{F}_j^i(\vec{C})\mathrm{d}t + \sqrt{2D}\mathrm{d}\vec{B}_t^{\,i}, \quad 1 \leqslant i \leqslant N_C, \quad 1 \leqslant j \leqslant N_V.$$

where $\mathbf{F}_j^i$ describes the total interaction force on vertex $\vec{v}_j^{\,i}$ caused by the current cell system $\vec{C}$ and $\sqrt{2D}\mathrm{d}\vec{B}_t^{\,i}$ models the two dimensional standard Brownian motion of cell i with diffusion coefficient D. Note, that all vertices of cell i perform the same Brownian motion such that the whole cell i moves in the direction of $\mathrm{d}\vec{B}_t^{\,i}$.

The simulation domain is always a square around the origin that is defined by $L > 0$ via

$$\Omega_L = [-L, L]^2.$$

How the interaction force $\mathbf{F}$ can be modelled will be shown the next chapter.

4 DF model dynamics

We characterise the interaction force $\mathbf{F}$ as the sum of gradient flows of energies.
A gradient flow describes how a system changes over time in a way that always reduces a given energy $E(\vec{C})$. To obtain the gradient flow of this energy on vertex $\vec{v}$, we must add the term $-\nabla_{\vec{v}} E(\vec{C})$ to $\mathbf{F}$. Since all our energy terms are positive, the lowest possible value is zero. So, the gradient flow moves the system step by step toward this minimum, always trying to decrease the energy until, ideally, it reaches zero. This is how we guide the motion of our cells: by letting them follow the gradient flow of each energy so that their shapes and vertex positions gradually adjust to reduce the total energy.
In [Vog23], the area, edge, interior angle, and overlap energies were introduced. The first three energies are responsible for maintaining the shape of each cell. All of these three according forces act on each cell in a vacuum based only on its own current cell shape.
Unlike in [Vog23], where each cell was assigned an individual desired state, we now assume a common desired state for all cells. This simplification allows for a more controlled analysis of the system's deformability and its influence on the collective dynamics. We assume that all cells are initially given in their desired states in order to prevent system instabilities right from the beginning.
Additionally, we introduce slight modifications to the energy formulation: rather than being defined locally on vertices or edges, the energies are now defined over entire cells. This adjustment provides a more coherent basis for deriving cell-level forces and ensures consistency with the global dynamic framework introduced in this study.
Interactions between different cells just arise from the overlap force, which acts to resolve overlaps and to prevent cell interpenetration. In the process of resolving overlaps, the shape of the cells will change. Once the overlap is resolved, the first three forces act to restore the cell's original shape.
The central question we aim to investigate in this thesis is how the deformability of individual cells influences the overall diffusivity of the cell system. But first, let us introduce each of the mentioned forces.
We define our energies as

$$E_k(x) = \frac{1}{k} |x_{\text{desired}} - x_{\text{current}}|^k,$$

where $k \in \mathbb{N}_{\geqslant 1}$ is a positive integer parameter specific to each energy term. Using different values of k allows us to model various types of energies and their corresponding forces, resulting in distinct dynamical behaviors that reflect different aspects of cell physics.
In order to compute the forces arising from these energy functions, we require the gradient ∇E. This leads us to compute derivatives of the form

$$\frac{\mathrm{d}}{\mathrm{d}x} |x|^k.$$

While $|x|^k$ is not classically differentiable at $x = 0$, it is weakly differentiable for all $k \in \mathbb{N}_{\geqslant 1}$. There exists a locally integrable function

$$x \mapsto k \operatorname{sgn}(x)|x|^{k-1} \in L^1_{loc}(\mathbb{R}),$$

such that for all $\phi \in C^\infty_C(\mathbb{R})$:

$$\begin{aligned}
\int_{\mathbb{R}} |x|^k \phi'(x) \mathrm{d}x &= \int_{\mathbb{R}_{\geqslant 0}} x^k \phi'(x) \mathrm{d}x + \int_{\mathbb{R}_{<0}} (-x)^k \phi'(x) \mathrm{d}x \\
&= [x^k \phi(x)]_0^\infty - \int_{\mathbb{R}_{\geqslant 0}} k x^{k-1} \phi(x) \mathrm{d}x + [(-x)^k \phi(x)]_0^\infty - \int_{\mathbb{R}_{<0}} k(-x)^{k-1} \phi(x) \mathrm{d}x \\
&= - \int_{\mathbb{R}_{\geqslant 0}} k x^{k-1} \phi(x) \mathrm{d}x - \int_{\mathbb{R}_{<0}} k(-x)^{k-1} \phi(x) \mathrm{d}x \\
&= - \int_{\mathbb{R}} k \operatorname{sgn}(x)|x|^{k-1} \phi(x) \mathrm{d}x.
\end{aligned}$$

Thus, $x \mapsto k \operatorname{sgn}(x)|x|^{k-1}$ is the weak derivative of $x \mapsto |x|^k$. We will use this weak derivative for all of our force computations.

4.1 Area force

The area force is designed to maintain each cell's area close to a preferred target value. In order to compute a cells area, which is the area of a positively orientated polygon, we can use the Shoelace formula from [Sho14].

Proposition 4.1. ***Shoelace formula for DF cells***
Let $C = (\vec{v}_1, \ldots, \vec{v}_N)$ be a DF cell with $\vec{v}_j = (v_j^x, v_j^y)^T$ for $j = 1, \ldots, N$. We determine the area A_C of C by applying the Shoelace formula

$$A_C = \tfrac{1}{2} \sum_{j=1}^{N} (v_j^x v_{j+1}^y - v_{j+1}^x v_j^y),$$

where $\vec{v}_{N+1} = \vec{v}_1$.
Proof.
An illustration supporting the proof is provided in 7, which is where the idea of the proof comes from. Without loss of generality, we may assume that all coordinates are positive. If this is not initially the case, the entire polygon can be translated into the positive quadrant without affecting its area.
For each $1 \leqslant j \leqslant N$ the edge $\overline{\vec{v}_j \, \vec{v}_{j+1}}$ is associated with the area T_j of the trapeze that arises when connecting the line segment vertically with the x axis. The signed trapeze area of T_j can be computed with

$$T_j = \tfrac{1}{2}(v_j^y + v_{j+1}^y)(v_j^x - v_{j+1}^x).$$

The area T_j has a positive sign if $v_j^x \geqslant v_{j+1}^x$ (green arrow in Figure 7) and a negative sign otherwise (red arrow). As depicted in the figure, the negatively signed areas precisely cancel the excess portions that would result from summing only the positively signed trapezoids. Thus the total polygon's area is equal to the sum of all trapezes

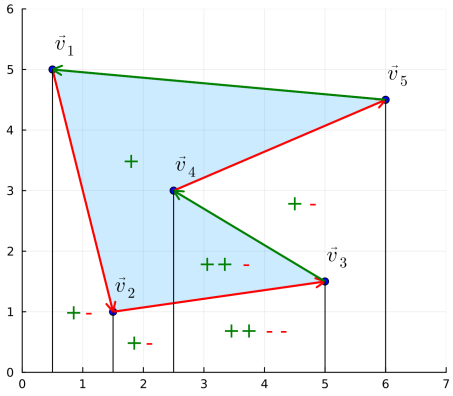

Figure 7: This figure shows a geometrical interpretation of the shoelace formula. The green arrows, which point from right to left, represent positive trapezoidal areas that contribute positively to the total area of the polygon. In contrast, the red arrows point from left to right and represent negative areas that are subtracted in the computation. The vertical black lines divide the plot into subregions. Within each subregion, green arrows are counted with plus signs and red arrows with minus signs. We observe that the subregions lying outside the polygon contain an equal number of plus and minus signs, indicating that their net contribution to the area is zero. In contrast, the subregions inside the polygon always have one more plus sign than minus signs, meaning their area is counted exactly once in the total. Overall, this illustrates that the method correctly computes the area of the polygon. Source: [Sho22]

$$A_C = \sum_{j=1}^{N} T_j = \tfrac{1}{2} \sum_{j=1}^{N} (v_j^y + v_{j+1}^y)(v_j^x - v_{j+1}^x) = \tfrac{1}{2} \sum_{j=1}^{N} (v_j^x v_{j+1}^y - v_{j+1}^x v_j^y).$$

□

With the Shoelace formula we are able to easily compute all cell areas at all times in the simulation. This enables us to implement the gradient flow over the area energy.

Definition 4.2. Area energy
The energy $A_k : (\mathbb{R}^2)^{N_V} \to \mathbb{R}_{\geqslant 0}$ for $k \in \mathbb{N}_{\geqslant 1}$, used to keep the cells at a constant volume, reads

$$A_k(C) = \frac{1}{k} |A_C - A_d|^k, \tag{5}$$

where A_d is the desired cell area of all cells and A_C is the current area of cell C.

To maintain the cell area during the simulation, we evaluate the gradient flow of the area energy which indicates the direction of motion for each vertex for preserving the cell area.

Proposition 4.3. ***Area force***
The area force $F_k^{(A)} : (\mathbb{R}^2)^{N_V} \to (\mathbb{R}^2)^{N_V}$ *that gets applied on cell* C *is given by*

$$F_k^{(A)}(C) = -(\nabla_{\vec{v}_1} A_k(C), \dots, \nabla_{\vec{v}_{N_V}} A_k(C))^T,$$

where the gradient $\nabla_{\vec{v}_j} A_k(C)$ *with respect to* $\vec{v}_j = (v_j^x, v_j^y)^T$ *is given by*

$$\nabla_{\vec{v}_j} A_k(C) = \frac{1}{2} \operatorname{sgn}(A_C - A_d) |A_C - A_d|^{k-1} \begin{pmatrix} v_{j+1}^y - v_{j-1}^y \\ v_{j-1}^x - v_{j+1}^x \end{pmatrix}, \tag{6}$$

for all $1 \leqslant j \leqslant N_V$.

Proof.
Choose $1 \leqslant j \leqslant N_V$.

$$\begin{aligned}
\nabla_{\vec{v}_j} A_k(C) &= \frac{1}{k} \nabla_{\vec{v}_j} |A_C - A_d|^k \\
&= \operatorname{sgn}(A_C - A_d) |A_C - A_d|^{k-1} \nabla_{\vec{v}_j} (A_C - A_d) \\
&= \operatorname{sgn}(A_C - A_d) |A_C - A_d|^{k-1} \nabla_{\vec{v}_j} A_C \\
&= \operatorname{sgn}(A_C - A_d) |A_C - A_d|^{k-1} \nabla_{\vec{v}_j} \left(\frac{1}{2} \sum_{k=1}^{N} (v_k^x v_{k+1}^y - v_{k+1}^x v_k^y) \right) \\
&= \frac{1}{2} \operatorname{sgn}(A_C - A_d) |A_C - A_d|^{k-1} \begin{pmatrix} \partial_{v_j^x} (v_j^x v_{j+1}^y - v_j^x v_{j-1}^y) \\ \partial_{v_j^y} (v_{j-1}^x v_j^y - v_{j+1}^x v_j^y) \end{pmatrix} \\
&= \frac{1}{2} \operatorname{sgn}(A_C - A_d) |A_C - A_d|^{k-1} \begin{pmatrix} v_{j+1}^y - v_{j-1}^y \\ v_{j-1}^x - v_{j+1}^x \end{pmatrix}
\end{aligned}$$

Remember that A_d *is just an independent constant.* □

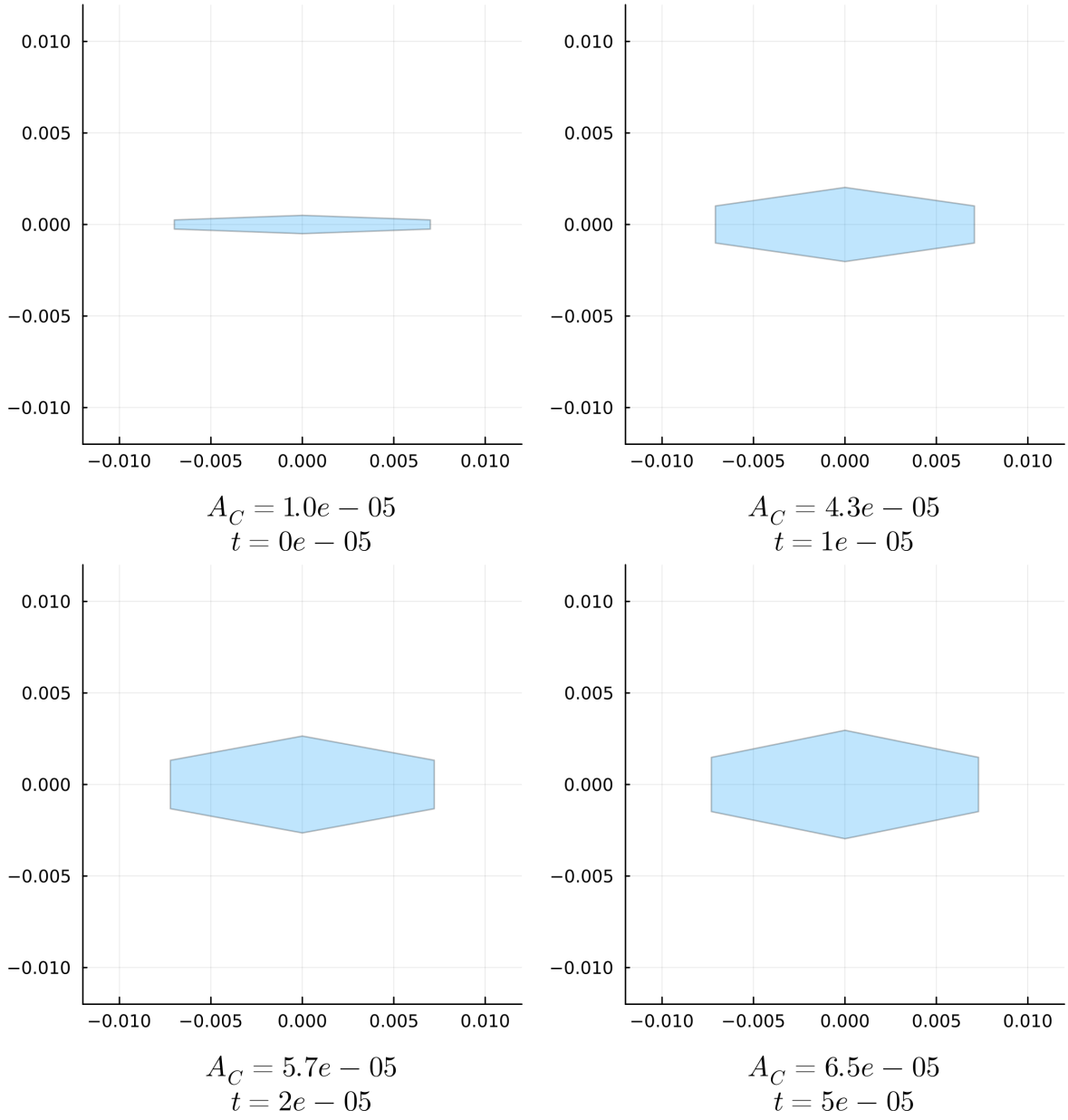

Figure 8: The top four plots show the evolution of a DF cell influenced solely by the area force, with $k = 2$ applied to the vertices and a force scaling of 4×10^8, at times $t \in \{0, 1 \times 10^{-5}, 2 \times 10^{-5}, 5 \times 10^{-5}\}$.
Thus, we have $\frac{\mathrm{d}\vec{v}}{\mathrm{d}t} = -4 \times 10^8 \nabla_{\vec{v}} A_2(C)$ for all vertices. The initial cell area is $A_C = 1 \times 10^{-5}$, while the desired cell area is set to 6.5×10^{-5}.
We deliberately chose an irregular cell shape, since small vertical changes to the vertices lead to large changes in area. Click *here* to view the corresponding animation (GIF).
The area force successfully restores the desired cell area after 5 time steps, which is also reflected Figure 26.

It is also valid to write $F_j^{(A)}(\vec{C})$ instead of $F_j^{(A)}(C)$, since C is included in $\vec{C}$.
Figure 8 illustrates how the area force acts on a cell to either expand or contract it

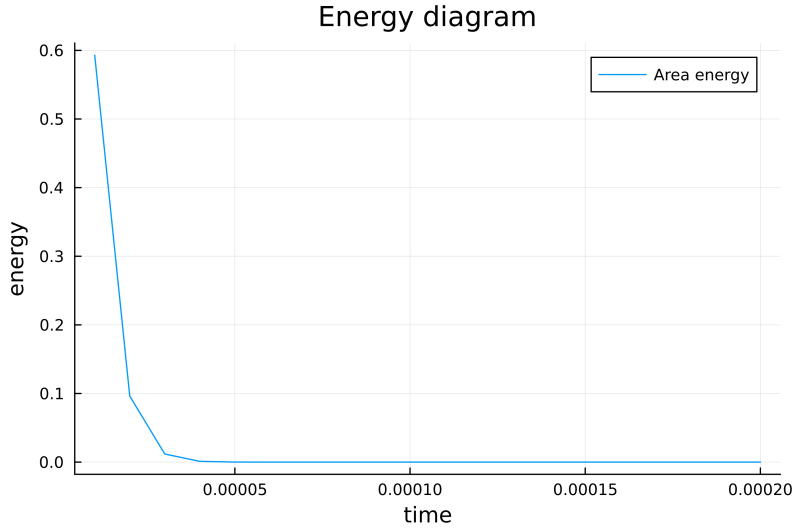

Figure 9: The area force successfully restores the desired cell area after 5 time steps as this energy diagram shows.

toward the desired target area.

4.2 Edge force

The next force we would like to model is the edge force. It acts on the cells' edges and aims to maintain their lengths. We define the edge $1 \leqslant j \leqslant N_V$ as

$$e_j = \overline{\vec{v}_j \, \vec{v}_{j+1}}$$

and we use the operator

$$E_C^j = \|\vec{v}_j - \vec{v}_{j+1}\|_2$$

to compute the length of the edge.
The according energy for this edge is:

Definition 4.4. Edge energy
The energy $E_k : (\mathbb{R}^2)^{N_V} \to \mathbb{R}_{\geqslant 0}$, used to keep the edges at a constant length, reads

$$E_k(C) = \sum_{j=1}^{N_V} \frac{1}{k} |E_C^j - E_d^j|^k, \tag{7}$$

where E_C^j is the current edge length and E_d^j is the desired edge length of edge j.
Since each vertex $\vec{v}_j$ influences exactly the edge lengths of the edges e_j and e_{j-1}, we get the total edge force on $\vec{v}_j$ with:

Proposition 4.5. ***Edge force***

The edge force $F_k^{(E)} : (\mathbb{R}^2)^{N_V} \to (\mathbb{R}^2)^{N_V}$ *that gets applied on cell* C *is given by*

$$F_k^{(E)}(C) = -(\nabla_{\vec{v}_1} E_k(C), \ldots, \nabla_{\vec{v}_{N_V}} E_k(C))^T,$$

where the gradient $\nabla_{\vec{v}_j} E_k(C)$ *with respect to* $\vec{v}_j = (v_j^x, v_j^y)^T$ *is given by*

$$\begin{aligned}\nabla_{\vec{v}_j} E_k(C) = \operatorname{sgn}(E_C^{j-1} - E_d^{j-1}) \frac{|E_C^{j-1} - E_d^{j-1}|^{k-1}}{E_C^{j-1}} \begin{pmatrix} v_j^x - v_{j-1}^x \\ v_j^y - v_{j-1}^y \end{pmatrix} \\ + \operatorname{sgn}(E_C^{j} - E_d^{j}) \frac{|E_C^{j} - E_d^{j}|^{k-1}}{E_C^{j}} \begin{pmatrix} v_j^x - v_{j+1}^x \\ v_j^y - v_{j+1}^y \end{pmatrix}\end{aligned} \tag{8}$$

for all $1 \leqslant j \leqslant N_V$.

Proof.

$$\nabla_{\vec{v}_j} E_k(C) = \nabla_{\vec{v}_j} \sum_{j=1}^{N_V} \frac{1}{k} |E_C^j - E_d^j|^k$$

$$\begin{aligned}\nabla_{\vec{v}_j} E_k(C) &= \nabla_{\vec{v}_j} \sum_{j=1}^{N_V} \frac{1}{k} |E_C^j - E_d^j|^k \\ &= \frac{1}{k} \nabla_{\vec{v}_j} |E_C^{j-1} - E_d^{j-1}|^k + \frac{1}{k} \nabla_{\vec{v}_j} |E_C^j - E_d^j|^k\end{aligned}$$

For the first summand, we can compute

$$\begin{aligned}
\frac{1}{k}\nabla_{\vec{v}_j}|E_C^{j-1}-E_d^{j-1}|^k &= \operatorname{sgn}(E_C^{j-1}-E_d^{j-1})|E_C^{j-1}-E_d^{j-1}|^{k-1}\nabla_{\vec{v}_j}(E_C^{j-1}-E_d^{j-1}) \\
&= \operatorname{sgn}(E_C^{j-1}-E_d^{j-1})|E_C^{j-1}-E_d^{j-1}|^{k-1}\nabla_{\vec{v}_j}E_C^{j-1} \\
&= \operatorname{sgn}(E_C^{j-1}-E_d^{j-1})|E_C^{j-1}-E_d^{j-1}|^{k-1}\cdot \\
&\qquad \nabla_{\vec{v}_j}\left[(v_{j-1}^x-v_j^x)^2+(v_{j-1}^y-v_j^y)^2\right]^{\frac{1}{2}} \\
&= \operatorname{sgn}(E_C^{j-1}-E_d^{j-1})|E_C^{j-1}-E_d^{j-1}|^{k-1}\cdot \\
&\qquad \left(\frac{1}{2E_C^{j-1}}\nabla_{\vec{v}_j}\left[(v_{j-1}^x-v_j^x)^2+(v_{j-1}^y-v_j^y)^2\right]\right) \\
&= \operatorname{sgn}(E_C^{j-1}-E_d^{j-1})|E_C^{j-1}-E_d^{j-1}|^{k-1}\cdot \\
&\qquad \nabla_{\vec{v}_j}\left[(v_{j-1}^x-v_j^x)^2+(v_{j-1}^y-v_j^y)^2\right]^{\frac{1}{2}} \\
&= \operatorname{sgn}(E_C^{j-1}-E_d^{j-1})|E_C^{j-1}-E_d^{j-1}|^{k-1}\cdot \\
&\qquad \left(\frac{1}{2E_C^{j-1}}\nabla_{\vec{v}_j}\left[(v_{j-1}^x-v_j^x)^2+(v_{j-1}^y-v_j^y)^2\right]\right) \\
&= \operatorname{sgn}(E_C^{j-1}-E_d^{j-1})\frac{|E_C^{j-1}-E_d^{j-1}|^{k-1}}{2E_C^{j-1}}\begin{pmatrix}\partial_{v_j^x}(v_{j-1}^x-v_j^x)^2\\ \partial_{v_j^y}(v_{j-1}^y-v_j^y)^2\end{pmatrix} \\
&= \operatorname{sgn}(E_C^{j-1}-E_d^{j-1})\frac{|E_C^{j-1}-E_d^{j-1}|^{k-1}}{2E_C^{j-1}}\begin{pmatrix}-2(v_{j-1}^x-v_j^x)\\ -2(v_{j-1}^y-v_j^y)\end{pmatrix} \\
&= \operatorname{sgn}(E_C^{j-1}-E_d^{j-1})\frac{|E_C^{j-1}-E_d^{j-1}|^{k-1}}{E_C^{j-1}}\begin{pmatrix}v_j^x-v_{j-1}^x\\ v_j^y-v_{j-1}^y\end{pmatrix}
\end{aligned}$$

For the second summand, we get:

$$\begin{aligned}
\frac{1}{k}\nabla_{\vec{v}_j}|E_C^j - E_d^j|^k &= \operatorname{sgn}(E_C^j - E_d^j)|E_C^j - E_d^j|^{k-1}\nabla_{\vec{v}_j}E_C^j \\
&= \operatorname{sgn}(E_C^j - E_d^j)|E_C^j - E_d^j|^{k-1}\cdot \\
&\qquad \left(\frac{1}{2E_C^j}\nabla_{\vec{v}_j}[(v_j^x - v_{j+1}^x)^2 + (v_j^y - v_{j+1}^y)^2]\right) \\
&= \operatorname{sgn}(E_C^j - E_d^j)|E_C^j - E_d^j|^{k-1}\cdot \\
&\qquad \left(\frac{1}{2E_C^j}\nabla_{\vec{v}_j}[(v_j^x - v_{j+1}^x)^2 + (v_j^y - v_{j+1}^y)^2]\right) \\
&= \operatorname{sgn}(E_C^j - E_d^j)\frac{|E_C^j - E_d^j|^{k-1}}{2E_C^j}\begin{pmatrix}\partial_{v_j^x}(v_j^x - v_{j+1}^x)^2 \\ \partial_{v_j^y}(v_j^y - v_{j+1}^y)^2\end{pmatrix} \\
&= \operatorname{sgn}(E_C^j - E_d^j)\frac{|E_C^j - E_d^j|^{k-1}}{2E_C^j}\begin{pmatrix}2(v_j^x - v_{j+1}^x) \\ 2(v_j^y - v_{j+1}^y)\end{pmatrix} \\
&= \operatorname{sgn}(E_C^j - E_d^j)\frac{|E_C^j - E_d^j|^{k-1}}{E_C^j}\begin{pmatrix}v_j^x - v_{j+1}^x \\ v_j^y - v_{j+1}^y\end{pmatrix}
\end{aligned}$$

□

An isolated application of the edge force can be seen in Figure 10.

4.3 Interior angle force

The combined application of the area and edge forces revealed instabilities in unfavorable configurations, where self-intersections of the cell edges occurred. Simulations without this energy sometimes can also result in constrictions at certain vertices, where the interior angle approaches 360°. To address this issue, we introduce the interior angle energy.
The first challenge is to consistently determine the interior angle at a given vertex throughout the simulation. Although we could apply the law of cosines and use arccos to compute the angle, this method would suffer from poor stability as the angle approaches 180°. A better alternative is to use the arctan2 function, as it remains reliably stable at all angles.

Definition 4.6. arctan2
The function

$$\operatorname{arctan2} : \mathbb{R}^2/\{0\} \to (-\pi, \pi]$$

is defined by:

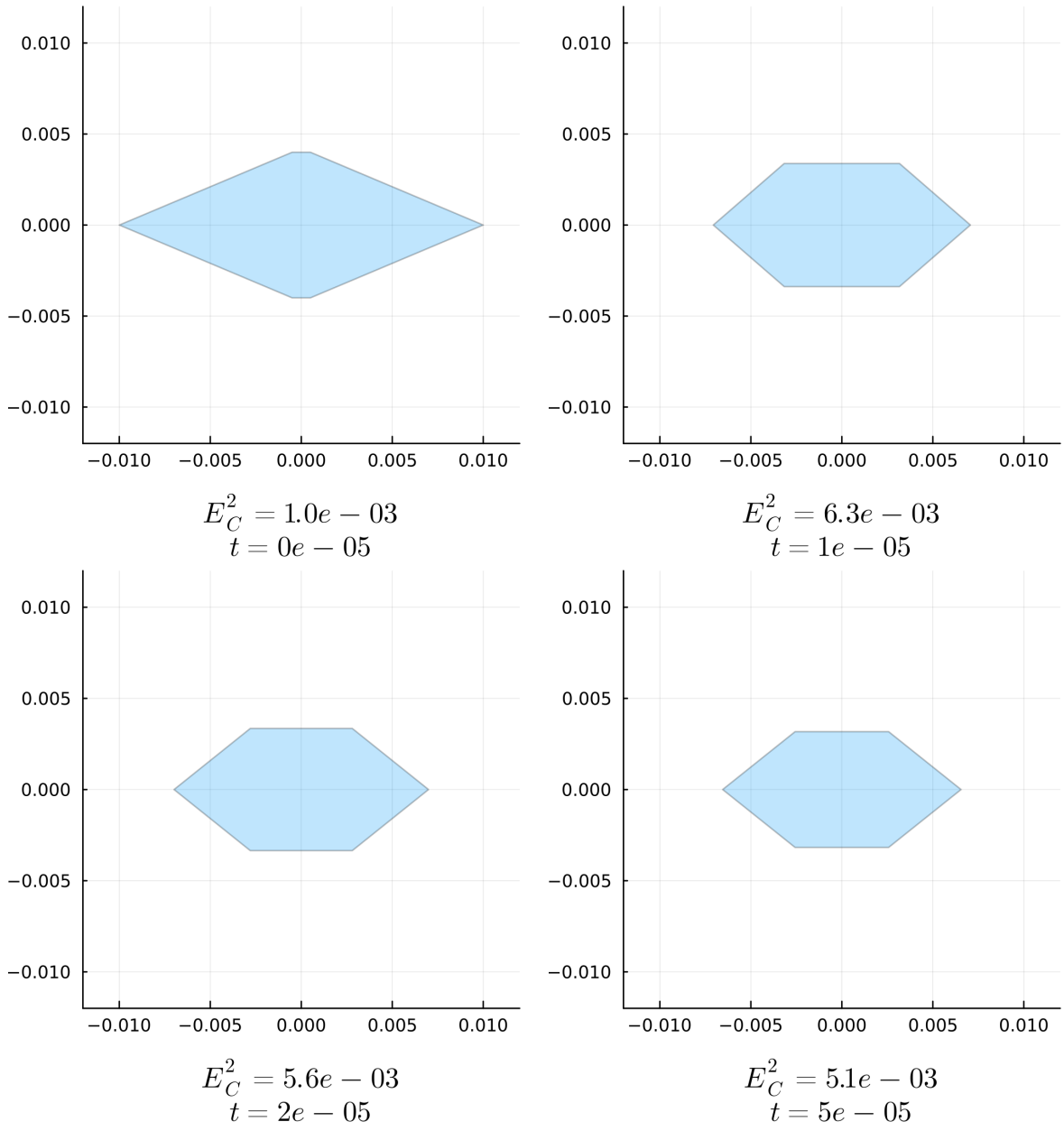

Figure 10: The top figures illustrate the evolution of a DF cell governed exclusively by the edge force, with $k = 2$ applied to the vertices and a force scaling of 3×10^4, at times $t \in \{0, 1 \times 10^{-5}, 2 \times 10^{-5}, 5 \times 10^{-5}\}$.
Accordingly, we have $\frac{\mathrm{d}\vec{v}}{\mathrm{d}t} = -3 \times 10^4 \nabla_{\vec{v}} E_2(C)$ for all vertices.
The initial edge length of the top edge is $E_C^2 = 1 \times 10^{-3}$, while the desired edge lengths are all set to 5×10^{-3}.
Click *here* to view the associated animation (GIF).
The edge force nearly restores the desired edge lengths after 5 time steps, which can also be observed in Figure 11.

$$\operatorname{arctan2}(\vec{v}) = \begin{cases} \arctan(\frac{v^y}{v^x}) & v^x > 0 \\ \arctan(\frac{v^y}{v^x}) + \pi & v^x < 0, v^y > 0 \\ \arctan(\frac{v^y}{v^x}) - \pi & v^x < 0, v^y < 0 \\ \pi & v^x < 0, v^y = 0 \\ \frac{\pi}{2} & v^x = 0, v^y > 0 \\ -\frac{\pi}{2} & v^x = 0, v^y < 0 \end{cases}.$$

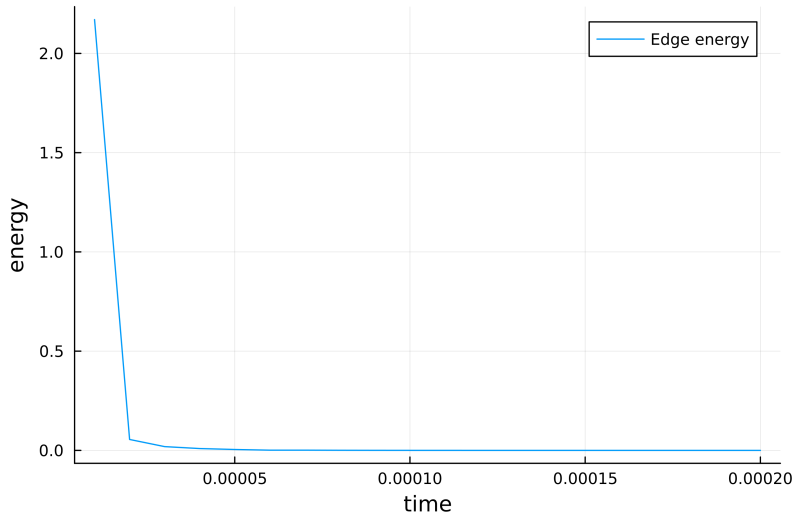

Figure 11: This energy diagram shows that the edge force nearly restores the desired edge lengths after 5 time steps.

The arctan2($\vec{v}$) function computes the angle of a vector $\vec{v}$ with respect to the positive x axis.
With this, we can compute the angles

$$\theta_1 = \text{arctan2}(\vec{v}_{j-1} - \vec{v_j}),$$
$$\theta_2 = \text{arctan2}(\vec{v}_{j+1} - \vec{v_j})$$

between the positive x axis and the vectors from $\vec{v}_j$ to its neighboring vertices $\vec{v}_{j-1}$ and $\vec{v}_{j+1}$. We get the searched angle at $\vec{v}_j$ by subtracting $\theta_1 - \theta_2$. To ensure that the angle lies within the interval $[0, 2\pi)$, we use the modulo operator $[\cdot]_{[0,2\pi)}$, which repeatedly adds or subtracts 2π from the angle until it falls within the desired range. Thus, our interior angle operator is:

$$I_C^j = [\text{arctan2}(\vec{v}_{j-1} - \vec{v_j}) - \text{arctan2}(\vec{v}_{j+1} - \vec{v_j})]_{[0,2\pi)}.$$

With that, we can define our interior angle energy.

Definition 4.7. Interior angle energy
The energy $I : (\mathbb{R}^2)^{N_V} \to \mathbb{R}_{\geqslant 0}$ associated with preserving the cell interior angles is given by

$$I_k(C) = \sum_{j=1}^{N_V} \frac{1}{k} |I_C^j - I_d^j|^k, \tag{9}$$

where I_d^j is the desired interior angle at vertex j and I_C^j is the current interior angle at vertex j of the considered cell.

We continue by computing the resulting force. The arctan2 function is partly defined and not truly differentiable. We still want to compute a gradient to use it for our interior angle force. It is

$$\text{arctan2}(\vec{v}) = \arctan\left(\frac{v^y}{v^x}\right) + \text{constant}$$

almost everywhere, just not on areas with measure zero. We just compute the gradient of $\arctan(\frac{v^y}{v^x})$ instead.
Another problem is the modulo operator $[\cdot]_{[0,2\pi)}$, which is not differentiable at the interval limits. However, we just neglect the modulo operator as it does not affect the dynamics of the gradient.

Proposition 4.8. ***Interior angle force***

The interior angle force $F_k^{(I)} : (\mathbb{R}^2)^{N_V} \to (\mathbb{R}^2)^{N_V}$ *that gets applied on cell* C *is given by*

$$F_k^{(I)}(C) = -(\nabla_{\vec{v}_1} I_k(C), \dots, \nabla_{\vec{v}_{N_V}} I_k(C))^T,$$

where the gradient $\nabla_{\vec{v}_j} I_k(C)$ *with respect to* $\vec{v}_j = (v_j^x, v_j^y)^T$ *is given by*

$$\begin{aligned}
\nabla_{\vec{v}_j} I_k(C) = {} & \operatorname{sgn}(I_C^{j-1} - I_d^{j-1}) |I_C^{j-1} - I_d^{j-1}|^{k-1} \left(-\frac{1}{\|\vec{v}_j - \vec{v}_{j-1}\|_2^2} \begin{pmatrix} v_{j-1}^y - v_j^y \\ v_j^x - v_{j-1}^x \end{pmatrix} \right) \\
& + \operatorname{sgn}(I_C^j - I_d^j) |I_C^j - I_d^j|^{k-1} \left(\frac{1}{\|\vec{v}_{j-1} - \vec{v}_j\|_2^2} \begin{pmatrix} v_{j-1}^y - v_j^y \\ v_j^x - v_{j-1}^x \end{pmatrix} \right. \\
& \left. - \frac{1}{\|\vec{v}_{j+1} - \vec{v}_j\|_2^2} \begin{pmatrix} v_{j+1}^y - v_j^y \\ v_j^x - v_{j+1}^x \end{pmatrix} \right) \\
& + \operatorname{sgn}(I_C^{j+1} - I_d^{j+1}) |I_C^{j+1} - I_d^{j+1}|^{k-1} \left(\frac{1}{\|\vec{v}_j - \vec{v}_{j+1}\|_2^2} \begin{pmatrix} v_{j+1}^y - v_j^y \\ v_j^x - v_{j+1}^x \end{pmatrix} \right)
\end{aligned} \tag{10}$$

for all $1 \leqslant j \leqslant N_V$.

Proof.
We are looking for

$$\nabla_{\vec{v}_j} I_k(C)$$

Vertex $\vec{v}_j$ *impacts the interior angles at* $\vec{v}_{j-1}$, $\vec{v}_j$ *and* $\vec{v}_{j+1}$. *Thus, we get*

$$\begin{aligned}
\nabla_{\vec{v}_j} I_k(C) &= \nabla_{\vec{v}_j} \sum_{j=1}^{N_V} \frac{1}{k} |I_d^j - I_C^j|^k \\
&= \nabla_{\vec{v}_j} \frac{1}{k} |I_d^{j-1} - I_C^{j-1}|^k + \nabla_{\vec{v}_j} \frac{1}{k} |I_d^j - I_C^j|^k + \nabla_{\vec{v}_j} \frac{1}{k} |I_d^{j+1} - I_C^{j+1}|^k
\end{aligned}$$

First, we will focus on the computation of $\nabla_{\vec{v}_j} \frac{1}{k} |I_d^j - I_C^j|^k$.

$$\begin{aligned}
\nabla_{\vec{v}_j} \frac{1}{k} |I_d^j - I_C^j|^k &= (I_d^j - I_C^j) \nabla_{\vec{v}_j} (-I_C^j) \\
&= (I_C^j - I_d^j) \nabla_{\vec{v}_j} I_C^j \\
&= (I_C^j - I_d^j) \nabla_{\vec{v}_j} [\operatorname{arctan2}(\vec{v}_{j-1} - \vec{v_j}) - \operatorname{arctan2}(\vec{v}_{j+1} - \vec{v_j})]_{[0,2\pi)}.
\end{aligned}$$

At this point, the previously mentioned simplifications come into play and we use $\arctan\left(\frac{v_{j-1}^y - v_j^y}{v_{j-1}^x - v_j^x}\right)$ *instead of* $\arctan2(\vec{v}_{j-1} - \vec{v_j})$ *and neglect the modulo operator. In the next step, we need to compute the gradient*

$$\nabla_{\vec{v}_j} \arctan\left(\frac{v_{j-1}^y - v_j^y}{v_{j-1}^x - v_j^x}\right).$$

Therefore, we define helper functions

$$f(\vec{v}) = \arctan\left(\frac{v^y}{v^x}\right)$$

and

$$g(\vec{v}_{j-1}, \vec{v}_j) = \begin{pmatrix} v_{j-1}^x - v_j^x \\ v_{j-1}^y - v_j^y \end{pmatrix}.$$

With these helper functions, we can write

$$\arctan\left(\frac{v_{j-1}^y - v_j^y}{v_{j-1}^x - v_j^x}\right) = (f \circ g)(\vec{v}_{j-1}, \vec{v}_j)$$

and use the two dimensional chain rule to stepwise compute the searched gradient.

$$\frac{\partial f(\vec{v})}{\partial v^x} = \frac{1}{1 + \left(\frac{v^y}{v^x}\right)^2}\left(-\frac{v^y}{(v^x)^2}\right) = -\frac{v^y}{(v^x)^2 + (v^y)^2}$$

$$\frac{\partial f(\vec{v})}{\partial v^y} = \frac{1}{1 + \left(\frac{v^y}{v^x}\right)^2}\frac{1}{v^x} = \frac{v^x}{(v^x)^2 + (v^y)^2}$$

$$\nabla_{v_j^x} g(\vec{v}_{j-1}, \vec{v}_j) = (-1, 0)^T$$
$$\nabla_{v_j^y} g(\vec{v}_{j-1}, \vec{v}_j) = (0, -1)^T$$

With that, we can compute:

$$\begin{aligned}
\frac{\partial(f \circ g(\vec{v}_{j-1}, \vec{v}_j))}{\partial v_j^x} &= (\nabla f \circ g(\vec{v}_{j-1}, \vec{v}_j))^T \cdot \nabla_{v_j^x} g(\vec{v}_{j-1}, \vec{v}_j) \\
&= \begin{pmatrix} -\frac{v_{j-1}^y - v_j^y}{(v_{j-1}^x - v_j^x)^2 + (v_{j-1}^y - v_j^y)^2} \\ \frac{v_{j-1}^x - v_j^x}{(v_{j-1}^x - v_j^x)^2 + (v_{j-1}^y - v_j^y)^2} \end{pmatrix}^T \cdot \begin{pmatrix} -1 \\ 0 \end{pmatrix} \\
&= \frac{v_{j-1}^y - v_j^y}{(v_{j-1}^x - v_j^x)^2 + (v_{j-1}^y - v_j^y)^2} \\
&= \frac{v_{j-1}^y - v_j^y}{\|\vec{v}_{j-1} - \vec{v}_j\|_2^2}
\end{aligned}$$

And similarly:

$$\begin{aligned}\frac{\partial(f \circ g(\vec{v}_{j-1}, \vec{v}_j))}{\partial v_j^y} &= (\nabla f \circ g(\vec{v}_{j-1}, \vec{v}_j))^T \cdot \nabla_{v_j^y} g(\vec{v}_{j-1}, \vec{v}_j) \\ &= \begin{pmatrix} -\frac{v_{j-1}^y - v_j^y}{(v_{j-1}^x - v_j^x)^2 + (v_{j-1}^y - v_j^y)^2} \\ \frac{v_{j-1}^x - v_j^x}{(v_{j-1}^x - v_j^x)^2 + (v_{j-1}^y - v_j^y)^2} \end{pmatrix}^T \cdot \begin{pmatrix} 0 \\ -1 \end{pmatrix} \\ &= -\frac{v_{j-1}^x - v_j^x}{(v_{j-1}^x - v_j^x)^2 + (v_{j-1}^y - v_j^y)^2} \\ &= \frac{v_j^x - v_{j-1}^x}{\|\vec{v}_{j-1} - \vec{v}_j\|_2^2}\end{aligned}$$

Overall, we get

$$\nabla_{\vec{v}_j} \arctan\left(\frac{v_{j-1}^y - v_j^y}{v_{j-1}^x - v_j^x}\right) = \frac{1}{\|\vec{v}_{j-1} - \vec{v}_j\|_2^2} \begin{pmatrix} v_{j-1}^y - v_j^y \\ v_j^x - v_{j-1}^x \end{pmatrix}.$$

Thus, we can come back to:

$$\begin{aligned}\nabla_{\vec{v}_j} \frac{1}{k} |I_d^j - I_C^j|^k &= (I_C^j - I_d^j) \nabla_{\vec{v}_j} \left(\arctan\left(\frac{v_{j-1}^y - v_j^y}{v_{j-1}^x - v_j^x}\right) - \arctan\left(\frac{v_{j+1}^y - v_j^y}{v_{j+1}^x - v_j^x}\right)\right) \\ &= (I_C^j - I_d^j) \Bigg(\frac{1}{\|\vec{v}_{j-1} - \vec{v}_j\|_2^2} \begin{pmatrix} v_{j-1}^y - v_j^y \\ v_j^x - v_{j-1}^x \end{pmatrix} \\ &\quad - \frac{1}{\|\vec{v}_{j+1} - \vec{v}_j\|_2^2} \begin{pmatrix} v_{j+1}^y - v_j^y \\ v_j^x - v_{j+1}^x \end{pmatrix} \Bigg).\end{aligned}$$

For the neighboring vertices, we need:

$$\nabla_{\vec{v}_j} \arctan\left(\frac{v_j^y - v_{j-1}^y}{v_j^x - v_{j-1}^x}\right) \text{ and } \nabla_{\vec{v}_j} \arctan\left(\frac{v_j^y - v_{j+1}^y}{v_j^x - v_{j+1}^x}\right).$$

Therefore, we can use the same helper function

$$f(\vec{v}) = \arctan\left(\frac{v^y}{v^x}\right),$$

but we need different functions for g, as the arrangement of the vertex coordinates differs a bit. We introduce

$$g_-(\vec{v}_{j-1}, \vec{v}_j) = \begin{pmatrix} v_j^x - v_{j-1}^x \\ v_j^y - v_{j-1}^y \end{pmatrix} = -g(\vec{v}_{j-1}, \vec{v}_j).$$

and

$$g_+(\vec{v}_j, \vec{v}_{j+1}) = \begin{pmatrix} v_j^x - v_{j+1}^x \\ v_j^y - v_{j+1}^y \end{pmatrix}.$$

The gradients are

$$\nabla_{v_j^x} g_-(\vec{v}_{j-1}, \vec{v}_j) = (1,0)^T, \nabla_{v_j^y} g_-(\vec{v}_{j-1}, \vec{v}_j) = (0,1)^T,$$

$$\nabla_{v_j^x} g_+(\vec{v}_{j-1}, \vec{v}_j) = (1,0)^T, \nabla_{v_j^y} g_+(\vec{v}_{j-1}, \vec{v}_j) = (0,1)^T.$$

Thus, the Jacobian of both g_- and g_+ is the identity matrix and we can just neglect it in the following chain rules. For the previous vertex, we get

$$\begin{aligned}
\nabla_{\vec{v}_j} \frac{1}{k} |I_C^{j-1} - I_d^{j-1}|^k &= \operatorname{sgn}(I_C^{j-1} - I_d^{j-1}) |I_C^{j-1} - I_d^{j-1}|^{k-1} \cdot \\
&\qquad \nabla_{\vec{v}_j} \left(\arctan\left(\frac{v_{j-2}^y - v_{j-1}^y}{v_{j-2}^x - v_{j-1}^x} \right) - \arctan\left(\frac{v_j^y - v_{j-1}^y}{v_j^x - v_{j-1}^x} \right) \right) \\
&= \operatorname{sgn}(I_C^{j-1} - I_d^{j-1}) |I_C^{j-1} - I_d^{j-1}|^{k-1} \cdot \\
&\qquad \nabla_{\vec{v}_j} \left(- \arctan\left(\frac{v_j^y - v_{j-1}^y}{v_j^x - v_{j-1}^x} \right) \right) \\
&= \operatorname{sgn}(I_C^{j-1} - I_d^{j-1}) |I_C^{j-1} - I_d^{j-1}|^{k-1} \cdot \\
&\qquad (-\nabla_{\vec{v}_j} (f \circ g_-(\vec{v}_{j-1}, \vec{v}_{j-1}))) \\
&= \operatorname{sgn}(I_C^{j-1} - I_d^{j-1}) |I_C^{j-1} - I_d^{j-1}|^{k-1} \cdot \\
&\qquad (-(\nabla f) \circ g_-(\vec{v}_{j-1}, \vec{v}_{j-1})) \\
&= \operatorname{sgn}(I_C^{j-1} - I_d^{j-1}) |I_C^{j-1} - I_d^{j-1}|^{k-1} \cdot \\
&\qquad \left(-\frac{1}{\|\vec{v}_j - \vec{v}_{j-1}\|_2^2} (v_{j-1}^y - v_j^y, v_j^x - v_{j-1}^x)^T \right).
\end{aligned}$$

Finally, for the successor vertex, we get

$$\begin{aligned}
\nabla_{\vec{v}_j} \frac{1}{k} |I_C^{j+1} - I_d^{j+1}|^k &= \operatorname{sgn}(I_C^{j+1} - I_d^{j+1}) |I_C^{j+1} - I_d^{j+1}|^{k-1} \cdot \\
&\qquad \nabla_{\vec{v}_j} \left(\arctan\left(\frac{v_j^y - v_{j+1}^y}{v_j^x - v_{j+1}^x} \right) - \arctan\left(\frac{v_{j+2}^y - v_{j+1}^y}{v_{j+2}^x - v_{j+1}^x} \right) \right) \\
&= \operatorname{sgn}(I_C^{j+1} - I_d^{j+1}) |I_C^{j+1} - I_d^{j+1}|^{k-1} \cdot \\
&\qquad \nabla_{\vec{v}_j} \left(\arctan\left(\frac{v_j^y - v_{j+1}^y}{v_j^x - v_{j+1}^x} \right) \right) \\
&= \operatorname{sgn}(I_C^{j+1} - I_d^{j+1}) |I_C^{j+1} - I_d^{j+1}|^{k-1} \cdot \\
&\qquad \nabla_{\vec{v}_j} \left(f \circ g_+(\vec{v}_{j-1}, \vec{v}_{j-1}) \right) \\
&= \operatorname{sgn}(I_C^{j+1} - I_d^{j+1}) |I_C^{j+1} - I_d^{j+1}|^{k-1} \cdot \\
&\qquad (\nabla f) \circ g_+(\vec{v}_{j-1}, \vec{v}_{j-1}) \\
&= \operatorname{sgn}(I_C^{j+1} - I_d^{j+1}) |I_C^{j+1} - I_d^{j+1}|^{k-1} \cdot \\
&\qquad \left(\frac{1}{\|\vec{v}_j - \vec{v}_{j+1}\|_2^2} (v_{j+1}^y - v_j^y, v_j^x - v_{j+1}^x)^T \right).
\end{aligned}$$

□

Figure 12 illustrates the isolated effect of the interior angle force.

4.4 Overlap force

Unlike the previous energies, which act independently on each cell, the overlap force is the first to account for interactions between multiple cells, thereby introducing cell-to-cell interaction into the simulation.

Deforming overlap force

The first overlap force, that we want to introduce, is an adapted form of the overlap force introduced in [Vog23]. It degenerates a cell overlap by influencing that cell shapes of the affected cells.
The challenging aspect of computing the overlap force lies in detecting overlaps within the cell system.

Definition 4.9. Overlap cell
An overlap cell between two DF cells C_i and C_m is a DF cell in the sence of Definition 3.2, composed of all vertices of C_i that lie inside C_m, all vertices of C_m that lie inside C_i and the intersection points of the cell walls of C_i and C_m.

Each pair of cells can have more than one overlap cell in unfavourable configurations. To be able to catch the correct dynamic in all cases, we need to introduce the set of all overlaps between a pair of DF cells.

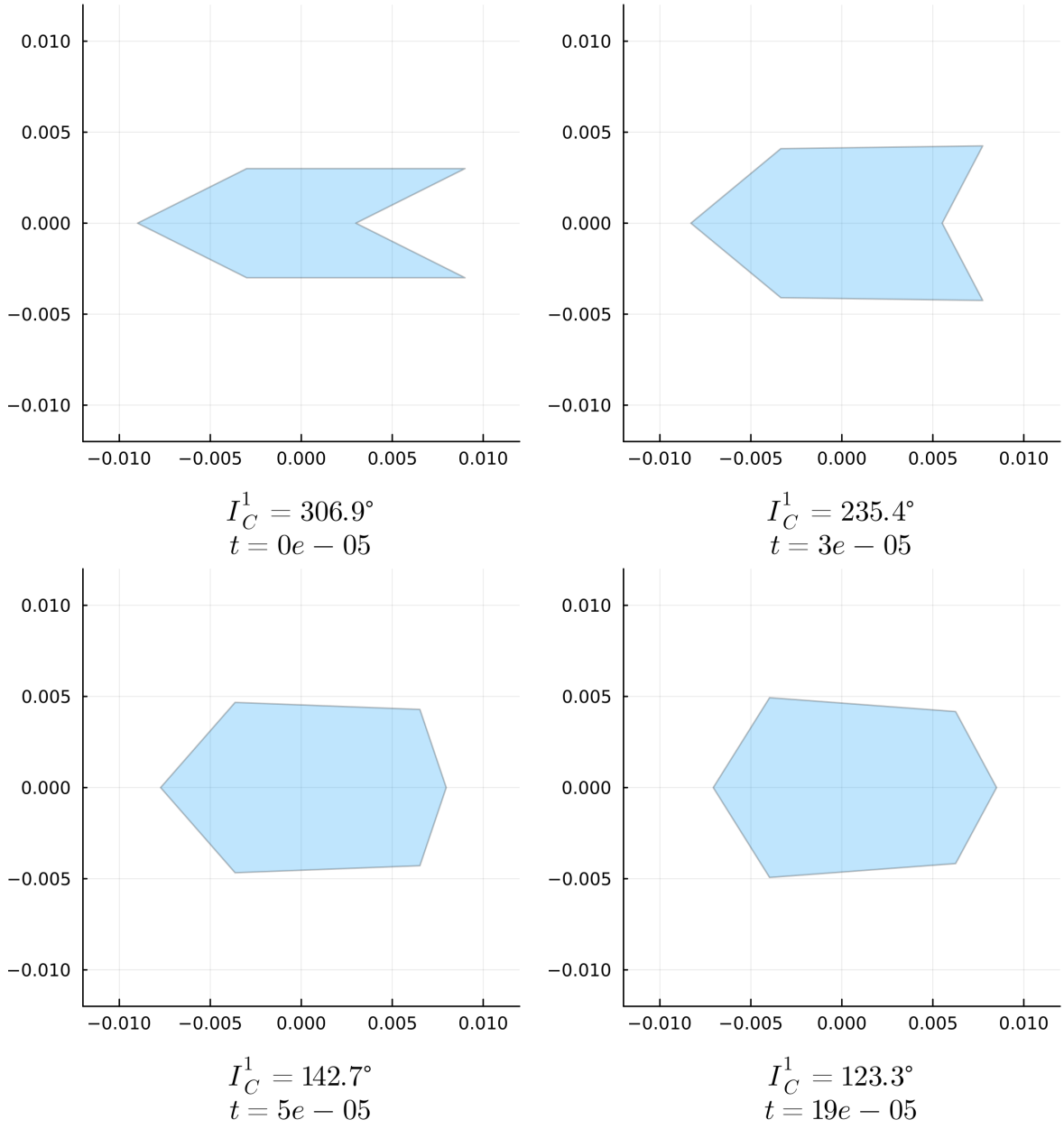

Figure 12: The top four plots depict the evolution of a DF cell subject only to the interior angle force with $k = 2$ applied to the vertices and a force scaling of 1×10^{-1}, at times $t \in \{0, 3 \times 10^{-5}, 5 \times 10^{-5}, 19 \times 10^{-5}\}$.
In this case, we have $\frac{\mathrm{d}\vec{v}}{\mathrm{d}t} = -1 \times 10^{-1} \nabla_{\vec{v}} I_2(C)$ for all vertices.
At the vertex with initial position $\vec{v}_1 = (0.003, 0.0)^T$, the starting interior angle is 306.9, while all desired interior angles are 120.
Click *here* to view the corresponding animation (GIF).
As seen in Figure 13, the interior angle force requires more time to reach the target configuration. This slower convergence is due to the reduced force scaling, which was necessary to avoid stability issues encountered at higher force scaling values.

Definition 4.10. Set of overlaps Ω_{C_i,C_m}

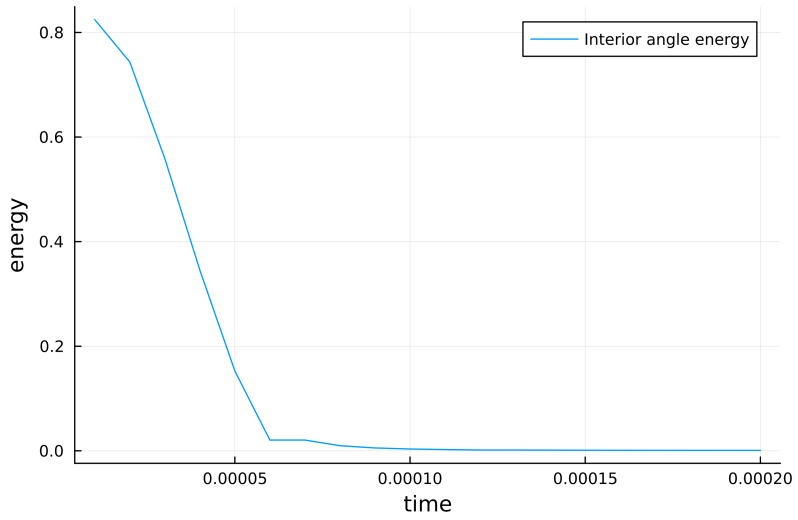

Figure 13: The interior angle force is able to minimize the interior angle energy over time.

Let $C_i, C_m \in \vec{C}$ be two DF cells. Then, the set of overlaps Ω_{C_i,C_m} is defined as

$$\Omega_{C_i,C_m} = \{\, D \mid D \text{ is an overlap cell formed between } C_i \text{ and } C_m \,\}.$$

Once all overlaps have been identified, we apply a dynamic similar to that of the area force, but with a desired area of zero. This generates a force that acts to eliminate the overlap by reducing its area to zero. The resulting force is then applied to the vertices of the original cells that define the overlapping region.
The first step in detecting overlaps is identifying the intersection points between cell boundaries. Intersections can be identified by representing the cell edges as line segments and computing the intersection points between segments belonging to different cells.
Having found all intersections, we can apply the following algorithm, that can be used to compute all overlaps between two cells.

Algorithm 4.11. ***Computation of a discrete overlap***

INPUT:

- *Discrete cells C and ζ*
- *List I of unused intersections of C and ζ*

function CONSTRUCTOVERLAP*(C, ζ, I)*
usedIntersections = List{Intersection}(I[1])
newOverlap = List{Vertices}(I[1])
currentIntersection = I[1]
for *counter = 1 : length(I)* ***do***
if *counter is even* ***then***
newPath, newIntersection = findPath(currentIntersection, C, I)
else
newPath, newIntersection = findPath(currentIntersection, ζ, I)

```
        end if
        append!(newOverlap, newPath)
        if newIntersection == I[1] then
            return newOverlap, usedIntersections
        else
            append!(newOverlap, newIntersection)
            append!(usedIntersections, newIntersection)
            currentIntersection = newIntersection
        end if
    end for
end function
```

OUTPUT:

- *A single intersection 'newOverlap' which occurs between C and ζ and which uses vertices from C and ζ as well as only intersections from I*
- *A list 'usedIntersections' of all intersection that are used in 'newOverlap'*

The algorithm begins by selecting the first intersection point $I[1]$ from the list I as the initial vertex of the overlap cell 'newOverlap'. This point is also added to the list 'usedIntersections'
Next, the function 'getOverlap' calls another function, 'findPath', which determines the path along the discrete cell ζ from the current intersection point to the next intersection in I encountered while traversing the edges of ζ. This next intersection is also returned by the function. The identified path is a list of vertices in ζ that lie strictly between the two intersections. It may be empty if the next intersection occurs on the same edge as the current one. Both the path and the newly found intersection are appended to 'newOverlap', and the intersection is also added to the list usedIntersections.
Since each intersection implies changing the cell from which the overlapping cell uses the edges, 'findPath' is now applied to the other cell. Again, it will deliver the next intersection as well as a list of the in between laying vertices. The vertex list always gets appended to 'newOverlap'.
If the newly found intersection is equal to the initial intersection $I[1]$, then the construction of the discrete overlap cell 'newOverlap' is complete. At this point, both 'newOverlap' and 'usedIntersections' can be returned by the function 'constructOverlap'.
Otherwise, the newly found intersection is appended to both 'newOverlap' and 'usedIntersections', and the process continues by calling 'findPath' on the other discrete cell. This step is repeated until the starting intersection is reached, completing the overlap cell construction.
Once an overlap between C and ζ has been successfully extracted, all intersections used in its construction can be removed from the list I, since each intersection point belongs to exactly one overlap. As long as I is not empty, the function 'constructOverlap' can be called again with the updated list to extract the next overlap. When I is empty, we can be certain that all intersections between C and ζ have been processed, and thus all overlaps between the two cells have been identified.

Each time 'findPath' is called, it is not immediately clear in which direction the function should traverse the vertices of the given cell. However, the correct direction can be determined using the following approach.
Starting from the current intersection passed into the function, move a small distance in one direction along the edge of the given cell where the intersection is located. Next, check whether this new point lies within the boundaries of the other cell as well. If the point is found in both cells, the chosen direction is correct. If not, then the opposite direction must be used.
A simple method to determine whether a point lies inside a polygon is to draw a ray from the point to the outside of the polygon. The number of intersections between the ray and the polygon's edges determines the point's position. If the number of intersections is odd, the point is inside the polygon. If it is even, the point is outside the polygon.

After introducing the method for detecting overlaps, we can now define the overlap force, which acts on the cell vertices involved in an overlap. This force is first computed based on the geometry of the overlap and then distributed to the corresponding vertices of the original cells.

Definition 4.12. Overlap energy
Let C_i and C_m be two cells from the system $\vec{C}$ and Ω_{C_i,C_m} be the set of all overlaps that appear between C_i and C_m, like explained above. Then, the total overlap energy $O_k : (\mathbb{R}^{2N_V})^{N_C} \to \mathbb{R}$ of the cell system is given by the formula

$$O_k(\vec{C}) = \sum_{i=1}^{N_C} \left(\sum_{m=i+1}^{N_C} \left(\sum_{D \in \Omega_{C_i,C_m}} \frac{1}{k} |A_D|^k \right) \right), \tag{11}$$

where A_D is the area of the overlap D.
We define the inner bracket to be the overlap energy of the cell pair C_i and C_m

$$O_k^{i,m} : (\mathbb{R}^{2N_V})^2 \to \mathbb{R}$$
$$O_k^{i,m}(C_i, C_m) = \sum_{D \in \Omega_{C_i,C_m}} \tfrac{1}{k} |A_D|^k.$$

To decrease the overlap areas during the simulation, we evaluate the gradient flow of the area energy with a desired area of zero which indicates the direction of motion for each vertex for reducing the overlap areas.

Definition 4.13. Intersection point and adjacent vertices
The vertices of an overlap cell D can be divided into the vertices that are either in C_i in C_m, we call that set

$$V(D) = \{\vec{v} \in D \,|\vec{v} \in C_i \cup C_m\},$$

and into the vertices that are neither in C_i nor C_m, named

$$W(D) = D \backslash V(D).$$

All overlap vertices in $W(D)$ are intersections between the cells C_i and C_m.
Each intersection $\vec{w}$ is dependent on two vertices of each cell that limit the edges

that intersect, where the intersection point $\vec{w}$ arises.
We call those four vertices **adjacent** to the intersection $\vec{w}$. All intersection adjacent vertices will get an extra deforming overlap dynamic applied, since they influence the overlap area, and thus the overlap energy, via the intersection point they create. From each cell we get one vertex that is part of the overlap cell, called the inside vertex, and one vertex that is not part of the overlap, called the outside vertex. For each intersection $\vec{w} \in W(D)$, we call its adjacent vertices

$$\text{adj}(\vec{w}) = \{\vec{v}^i_{\text{in}}, \vec{v}^i_{\text{out}}, \vec{v}^m_{\text{in}}, \vec{v}^m_{\text{out}}\}.$$

In order to refer to the inside or outside vertices, we define the sets

$$\text{in}(\vec{w}) = \{\vec{v}^i_{\text{in}}, \vec{v}^m_{\text{in}}\} \text{ and } \text{out}(\vec{w}) = \{\vec{v}^i_{\text{out}}, \vec{v}^m_{\text{out}}\}.$$

Figure 14 illustrates what the in- and outside vertices of an intersection are.
Given the four adjacent vertices, we can always compute the intersection with the function:

$$w : (\mathbb{R}^2)^4 \to \mathbb{R}^2,$$
$$w(\vec{v}^i_{\text{out}}, \vec{v}^i_{\text{in}}, \vec{v}^m_{\text{out}}, \vec{v}^m_{\text{in}}) = \vec{v}^i_{\text{out}} + \frac{(\vec{v}^m_{\text{out}} - \vec{v}^i_{\text{out}}) \times (\vec{v}^m_{\text{in}} - \vec{v}^m_{\text{out}})}{(\vec{v}^i_{\text{in}} - \vec{v}^i_{\text{out}}) \times (\vec{v}^m_{\text{in}} - \vec{v}^m_{\text{out}})}(\vec{v}^i_{\text{in}} - \vec{v}^i_{\text{out}}),$$

where $(a^x, a^y)^T \times (b^x, b^y)^T = a^x b^y - a^y b^x$ denotes the two dimensional cross product.

Proposition 4.14. ***Partial derivatives of intersection points***
We use $w(\vec{v}^i_{out}, \vec{v}^i_{in}, \vec{v}^m_{out}, \vec{v}^m_{in})$ to compute the influence of the adjacent vertices to the intersection point via their partial derivatives. In this proposition, we compute the needed partial derivatives.
We define the following auxiliary terms:

$$f = (\vec{v}^m_{out} - \vec{v}^i_{out}) \times (\vec{v}^m_{in} - \vec{v}^m_{out})$$

$$g = (\vec{v}^i_{in} - \vec{v}^i_{out}) \times (\vec{v}^m_{in} - \vec{v}^m_{out})$$

$$t = \frac{f}{g}$$

$$w = \vec{v}^i_{out} + t(\vec{v}^i_{in} - \vec{v}^i_{out})$$

It is sufficient to just compute the partial derivatives with respect to $\vec{v}^i_{in}$ and $\vec{v}^i_{out}$. If we want the dynamic for the vertices of the other cell, we just switch the arrangement of the arguments (switch $\vec{v}^i_{in}$ with $\vec{v}^j_{in}$ and $\vec{v}^i_{out}$ with $\vec{v}^j_{out}$) and then use the same partial derivatives as for the first cell.

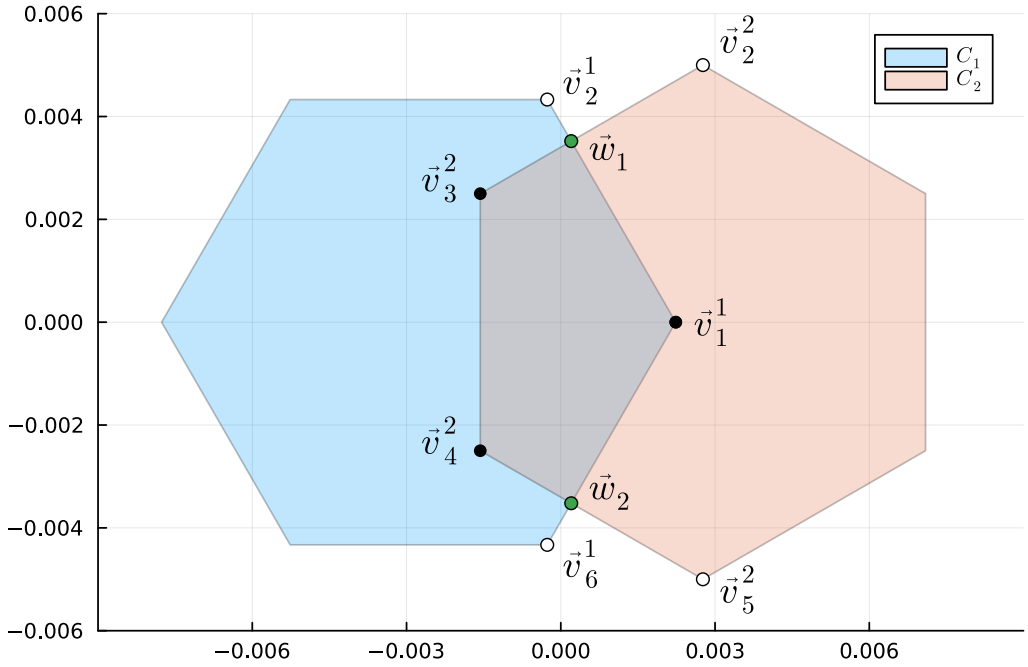

Figure 14: Here, we can see a DF cell setup with two cells having an overlap. The intersection points $\vec{w}_1$ and $\vec{w}_2$ are marked green. We can also see all inside vertices colored in black and the outside vertices colored in white. In this example, we have $\text{adj}(\vec{w}_1) = \{\vec{v}_1^{\,1}, \vec{v}_2^{\,1}, \vec{v}_2^{\,2}, \vec{v}_3^{\,2}\}$, $\text{in}(\vec{w}_1) = \{\vec{v}_1^{\,1}, \vec{v}_3^{\,2}\}$ and $\text{out}(\vec{w}_1) = \{\vec{v}_2^{\,1}, \vec{v}_2^{\,2}\}$ for the first intersection and $\text{adj}(\vec{w}_2) = \{\vec{v}_1^{\,1}, \vec{v}_6^{\,1}, \vec{v}_4^{\,2}, \vec{v}_5^{\,2}\}$, $\text{in}(\vec{w}_2) = \{\vec{v}_1^{\,1}, \vec{v}_4^{\,2}\}$ and $\text{out}(\vec{w}_2) = \{\vec{v}_6^{\,1}, \vec{v}_5^{\,2}\}$ for the second intersection.

The partial derivatives are:

$$D_{\vec{v}^i_{out}} w(\vec{v}^i_{out}, \vec{v}^i_{in}, \vec{v}^m_{out}, \vec{v}^m_{in}) : (\mathbb{R}^2)^4 \to \mathbb{R}^{2\times 2},$$

$$D_{\vec{v}^i_{out}} w(\vec{v}^i_{out}, \vec{v}^i_{in}, \vec{v}^m_{out}, \vec{v}^m_{in}) = (1-t)I_2 + \frac{g-f}{g^2}(\vec{v}^i_{in} - \vec{v}^i_{out}) \begin{pmatrix} -(v^{m,y}_{in} - v^{m,y}_{out}) \\ v^{m,x}_{in} - v^{m,x}_{out} \end{pmatrix}^T, \tag{12}$$

$$D_{\vec{v}^i_{in}} w(\vec{v}^i_{out}, \vec{v}^i_{in}, \vec{v}^m_{out}, \vec{v}^m_{in}) : (\mathbb{R}^2)^4 \to \mathbb{R}^{2\times 2},$$

$$D_{\vec{v}^i_{in}} w(\vec{v}^i_{out}, \vec{v}^i_{in}, \vec{v}^m_{out}, \vec{v}^m_{in}) = tI_2 \; + \; \frac{f}{g^2}(\vec{v}^i_{in} - \vec{v}^i_{out}) \begin{pmatrix} -(v^{m,y}_{in} - v^{m,y}_{out}) \\ v^{m,x}_{in} - v^{m,x}_{out} \end{pmatrix}^T. \tag{13}$$

Proof.

First of all, we compute the gradients of f and g:

$$\begin{aligned} \nabla_{\vec{v}^i_{out}} f &= \nabla_{\vec{v}^i_{out}} [(v^{m,x}_{out} - v^{i,x}_{out})(v^{m,y}_{in} - v^{m,y}_{out}) - (v^{m,y}_{out} - v^{i,y}_{out})(v^{m,x}_{in} - v^{m,x}_{out})] \\ &= \begin{pmatrix} -(v^{m,y}_{in} - v^{m,y}_{out}) \\ v^{m,x}_{in} - v^{m,x}_{out} \end{pmatrix}, \end{aligned}$$

$$\begin{aligned} \nabla_{\vec{v}^i_{in}} f &= \nabla_{\vec{v}^i_{in}} [(v^{m,x}_{out} - v^{i,x}_{out})(v^{m,y}_{in} - v^{m,y}_{out}) - (v^{m,y}_{out} - v^{i,y}_{out})(v^{m,x}_{in} - v^{m,x}_{out})] \\ &= \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \end{aligned}$$

$$\begin{aligned} \nabla_{\vec{v}^i_{out}} g &= \nabla_{\vec{v}^i_{out}} [(v^{i,x}_{in} - v^{i,x}_{out})(v^{m,y}_{in} - v^{m,y}_{out}) - (v^{i,y}_{in} - v^{i,y}_{out})(v^{m,x}_{in} - v^{m,x}_{out})] \\ &= \begin{pmatrix} -(v^{m,y}_{in} - v^{m,y}_{out}) \\ v^{m,x}_{in} - v^{m,x}_{out} \end{pmatrix}, \end{aligned}$$

$$\begin{aligned} \nabla_{\vec{v}^i_{in}} g &= \nabla_{\vec{v}^i_{in}} [(v^{i,x}_{in} - v^{i,x}_{out})(v^{m,y}_{in} - v^{m,y}_{out}) - (v^{i,y}_{in} - v^{i,y}_{out})(v^{m,x}_{in} - v^{m,x}_{out})] \\ &= \begin{pmatrix} v^{m,y}_{in} - v^{m,y}_{out} \\ -(v^{m,x}_{in} - v^{m,x}_{out}) \end{pmatrix}. \end{aligned}$$

Now, we can succeed with the gradients of t:

$$\begin{aligned}
\nabla_{\vec{v}_{out}^{\,i}} t &= \nabla_{\vec{v}_{out}^{\,i}} \frac{f}{g} \\
&= \frac{(\nabla_{\vec{v}_{out}^{\,i}} f)g - (\nabla_{\vec{v}_{out}^{\,i}} g)f}{g^2} \\
&= \frac{1}{g^2}\left(\begin{pmatrix} -(v_{in}^{m,y} - v_{out}^{m,y}) \\ v_{in}^{m,x} - v_{out}^{m,x} \end{pmatrix} g - \begin{pmatrix} -(v_{in}^{m,y} - v_{out}^{m,y}) \\ v_{in}^{m,x} - v_{out}^{m,x} \end{pmatrix} f\right) \\
&= \frac{g-f}{g^2}\begin{pmatrix} -(v_{in}^{m,y} - v_{out}^{m,y}) \\ v_{in}^{m,x} - v_{out}^{m,x} \end{pmatrix}
\end{aligned}$$

$$\begin{aligned}
\nabla_{\vec{v}_{in}^{\,i}} t &= \nabla_{\vec{v}_{in}^{\,i}} \frac{f}{g} \\
&= \frac{(\nabla_{\vec{v}_{in}^{\,i}} f)g - (\nabla_{\vec{v}_{in}^{\,i}} g)f}{g^2} \\
&= \frac{1}{g^2}\left(-\begin{pmatrix} v_{in}^{m,y} - v_{out}^{m,y} \\ -(v_{in}^{m,x} - v_{out}^{m,x}) \end{pmatrix} f\right). \\
&= \frac{f}{g^2}\begin{pmatrix} -(v_{in}^{m,y} - v_{out}^{m,y}) \\ v_{in}^{m,x} - v_{out}^{m,x} \end{pmatrix}.
\end{aligned}$$

And finally, we can compute the partial derivatives of $w = (w_1, w_2)^T$*:*

$$\frac{\partial w_1}{\partial v_{out}^{i,x}} = 1 + \frac{\partial t}{\partial v_{out}^{i,x}}(v_{in}^{i,x} - v_{out}^{i,x}) - t, \quad \frac{\partial w_1}{\partial v_{out}^{i,y}} = 0 + \frac{\partial t}{\partial v_{out}^{i,y}}(v_{in}^{i,x} - v_{out}^{i,x}) + 0,$$

$$\frac{\partial w_2}{\partial v_{out}^{i,x}} = 0 + \frac{\partial t}{\partial v_{out}^{i,x}}(v_{in}^{i,y} - v_{out}^{i,y}) + 0, \quad \frac{\partial w_2}{\partial v_{out}^{i,y}} = 1 + \frac{\partial t}{\partial v_{out}^{i,y}}(v_{in}^{i,y} - v_{out}^{i,y}) - t,$$

$$\begin{aligned}
\Rightarrow D_{\vec{v}_{out}^{\,i}} w &= (1-t)I_2 + (\vec{v}_{in}^{\,i} - \vec{v}_{out}^{\,i})(\nabla_{\vec{v}_{out}^{\,i}} t)^T \\
&= (1-t)I_2 + (\vec{v}_{in}^{\,i} - \vec{v}_{out}^{\,i}) \left(\frac{g-f}{g^2} \begin{pmatrix} -(v_{in}^{m,y} - v_{out}^{m,y}) \\ v_{in}^{m,x} - v_{out}^{m,x} \end{pmatrix} \right)^T \\
&= (1-t)I_2 + \frac{g-f}{g^2}(\vec{v}_{in}^{\,i} - \vec{v}_{out}^{\,i}) \begin{pmatrix} -(v_{in}^{m,y} - v_{out}^{m,y}) \\ v_{in}^{m,x} - v_{out}^{m,x} \end{pmatrix}^T,
\end{aligned}$$

$$\frac{\partial w_1}{\partial v_{in}^{i,x}} = \frac{\partial t}{\partial v_{in}^{i,x}}(v_{in}^{i,x} - v_{out}^{i,x}) + t, \quad \frac{\partial w_1}{\partial v_{in}^{i,y}} = \frac{\partial t}{\partial v_{in}^{i,x}}(v_{in}^{i,x} - v_{out}^{i,x}) + 0,$$

$$\frac{\partial w_2}{\partial v_{in}^{i,x}} = \frac{\partial t}{\partial v_{in}^{i,x}}(v_{in}^{i,y} - v_{out}^{i,y}) + 0, \quad \frac{\partial w_2}{\partial v_{in}^{i,y}} = \frac{\partial t}{\partial v_{in}^{i,y}}(v_{in}^{i,y} - v_{out}^{i,y}) + t,$$

$$\begin{aligned}
\Rightarrow D_{\vec{v}_{in}^{\,i}} w &= tI_2 + (\vec{v}_{in}^{\,i} - \vec{v}_{out}^{\,i})(\nabla_{\vec{v}_{in}^{\,i}} t)^T \\
&= tI_2 + (\vec{v}_{in}^{\,i} - \vec{v}_{out}^{\,i}) \left(\frac{f}{g^2} \begin{pmatrix} -(v_{in}^{m,y} - v_{out}^{m,y}) \\ v_{in}^{m,x} - v_{out}^{m,x} \end{pmatrix} \right)^T \\
&= tI_2 + \frac{f}{g^2}(\vec{v}_{in}^{\,i} - \vec{v}_{out}^{\,i}) \begin{pmatrix} -(v_{in}^{m,y} - v_{out}^{m,y}) \\ v_{in}^{m,x} - v_{out}^{m,x} \end{pmatrix}^T.
\end{aligned}$$

Proposition 4.15. ***Deforming overlap force***
Each overlap cell $D \in \Omega_{C_i,C_m}$ *is a list of vertices, that form the overlap, just like a DF cell.*

The deforming overlap gradient is then given by

$$\nabla_{\vec{v}_j^{\,i}} O_k(\vec{C}) = \sum_{D \in \Omega_{C_i,C_m}} |A_D|^{k-1} \Bigg(\mathbf{1}_{V(D)}(\vec{v}_j^{\,i}) \nabla_{\vec{v}_j^{\,i}} A(D) + \\ + \sum_{\vec{w} \in W(D)} \left(\mathbf{1}_{out(w)}(\vec{v}_j^{\,i}) D_{\vec{v}_{out}^{\,i}} \vec{w} + \mathbf{1}_{in(w)}(\vec{v}_j^{\,i}) D_{\vec{v}_{in}^{\,i}} \vec{w} \right) \nabla_{\vec{w}} A(D) \Bigg), \tag{14}$$

for all $1 \leqslant j \leqslant N_V$ and $1 \leqslant i \leqslant N_C$, where $out(w)$, $in(w) \subset adj(w)$ denote the sets of outside and inside overlap-adjacent vertices, respectively.
Note, that the Formulas 12 and 13 define $\frac{\partial \vec{w}}{\partial \vec{v}_{j,out}^{\,i}}$ and $\frac{\partial \vec{w}}{\partial \vec{v}_{j,in}^{\,i}}$, respectively.
The difference between $\nabla_{\vec{v}_j^{\,i}} A(D)$ and $\nabla_{\vec{w}} A(D)$ is that $\nabla_{\vec{v}_j^{\,i}} A(D)$ uses the neighboring overlap vertices of $\vec{v}_j^{\,i}$ itself in the Area Gradient Formula 6, whereas $\nabla_{\vec{w}} A(D)$ uses the neighbors of the corresponding intersection point in D, which is not the same overlap vertex as $\vec{v}_j^{\,i}$. A_D is the area of the overlap D.
The indicator function $\mathbf{1}_A(\vec{v})$ equals one, if $\vec{v} \in A$ and is zero otherwise.
The deforming overlap force $F_k^{(\hat{O})} : (\mathbb{R}^{2N_V})^{N_C} \to (\mathbb{R}^{2N_V})^{N_C}$ that gets applied on the whole cell system $\vec{C} = (C_1, \ldots, C_{N_C})$ is then given by

$$F_k^{(\hat{O})}(\vec{C}) = -(\nabla_{\vec{v}_1^{\,1}} O_k(\vec{C}), \ldots, \nabla_{\vec{v}_{N_V}^{\,1}} O_k(\vec{C}), \ \cdots, \vec{v}_1^{\,N_C} O_k(\vec{C}), \ldots, \nabla_{\vec{v}_{N_V}^{\,N_C}} O_k(\vec{C}))^T.$$

For addressing the overlap force that acts on cell $1 \leqslant i \leqslant N_C$, we define

$$F_{k,i}^{(\hat{O})} : (\mathbb{R}^{2N_V})^{N_C} \to (\mathbb{R}^{2N_V}),$$
$$F_{k,i}^{(\hat{O})}(\vec{C}) = -(\nabla_{\vec{v}_1^{\,i}} O_k(\vec{C}), \ldots, \nabla_{\vec{v}_{N_V}^{\,i}} O_k(\vec{C}))^T.$$

Proof.
Although we did not noted it like this before, we must be aware that A_D is actually dependent on all overlap vertices, e.g. $A_D = A(D)$. We will also use that notation in the coming computation. Since the area $A(D)$ is always positive, we can drop the absolute value.
We aim to compute

$$\begin{aligned} \nabla_{\vec{v}_j^{\,i}} O_k(\vec{C}) &= \nabla_{\vec{v}_j^{\,i}} \sum_{i=1}^{N_C} \left(\sum_{m=i+1}^{N_C} \left(\sum_{D \in \Omega_{C_i,C_m}} \frac{1}{k} A_D^k \right) \right) \\ &= \nabla_{\vec{v}_j^{\,i}} \sum_{m \neq i} O_k^{i,m}(C_i, C_m) \\ &= \nabla_{\vec{v}_j^{\,i}} \sum_{m \neq i} \sum_{D \in \Omega_{C_i,C_m}} \frac{1}{k} A(D)^k \\ &= \sum_{m \neq i} \sum_{D \in \Omega_{C_i,C_m}} \nabla_{\vec{v}_j^{\,i}} \frac{1}{k} A(D)^k. \end{aligned}$$

Now, there are different cases.
Case 1: *$\vec{v}_j^{\,i} \notin D$ and $\vec{v}_j^{\,i} \notin adj(\vec{w})$ $\forall \vec{w} \in W(D)$*

In the first case, the considered vertex is neither a vertex from the overlap cell D, nor adjacent to any intersection point.
Hence, this vertex has zero impact on the overlap and its gradient is zero:

$$\nabla_{\vec{v}_j^i} \frac{1}{k} A_D^k = 0.$$

Case 2: $\vec{v}_j^i \notin D$ *and* $\exists \vec{w} \in W(D) : \vec{v}_j^i \in adj(\vec{w})$
For case 2, we consider a vertex that is not directly an overlap vertex, but it influences the overlap cell by influencing an intersection point $\vec{w}$. This means that $\vec{v}_j^i$ is an outside adjacent vertex of the intersection $\vec{w}$. We compute:

$$\begin{aligned}\nabla_{\vec{v}_j^i} \frac{1}{k} A(D)^k &= A_D^{k-1} \nabla_{\vec{v}_j^i} A(D) \\ &= A_D^{k-1} \sum_{\vec{w} \in W(D)} \mathbf{1}_{out(w)}(\vec{v}_j^i) D_{\vec{v}_{out}^i} \vec{w} \nabla_{\vec{w}} A(D),\end{aligned}$$

where, according to Equation 12

$$D_{\vec{v}_{out}^i} \vec{w} = (1-t) I_2 + \frac{g-f}{g^2} (\vec{v}_{in}^i - \vec{v}_j^i) \begin{pmatrix} -(v_{in}^{i,y} - v_j^{i,y}) \\ v_{in}^{i,x} - v_j^{i,x} \end{pmatrix}^T,$$

because $\vec{v}_j^i$ is an outside vertex in this case and $\vec{v}_{in}^i$ is the vertex adjacent to $\vec{v}_j^i$ in cell i, such that these vertices build the edge causing the intersection.
The gradient $\nabla_{\vec{w}} A(D)$ can easily be computed via the Shoelace Formula 4.1, as in the area gradient from Formula 6:

$$\nabla_{\vec{w}} A(D) = \frac{1}{2} \begin{pmatrix} d_{j+1}^y - d_{j-1}^y \\ d_{j-1}^x - d_{j+1}^x \end{pmatrix},$$

where $\vec{d}_{j-1}^D = (d_{j-1}^x, d_{j-1}^y)^T$ and $\vec{d}_{j+1}^D = (d_{j+1}^x, d_{j+1}^y)^T$ are the neighboring vertices of the intersection $\vec{w}$ in the overlap D.

Case 3: $\vec{v}_j^i \in D$ *and* $\vec{v}_j^i \notin adj(\vec{w})\ \forall \vec{w} \in W(D)$
Now, $\vec{v}_j^i$ is a pure inside overlap vertex, in the sence that it does not have an intersection point as a neighbor in the overlap. This dynamic is quite easy, since we just have to use the Shoelace Formular 4.1 for once more:

$$\begin{aligned}\nabla_{\vec{v}_j^i} \frac{1}{k} A(D)^k &= A_D^{k-1} \nabla_{\vec{v}_j^i} A(D) \\ &= A_D^{k-1} \mathbf{1}_{V(D)}(\vec{v}_j^i) \frac{1}{2} \begin{pmatrix} d_{j+1}^y - d_{j-1}^y \\ d_{j-1}^x - d_{j+1}^x \end{pmatrix},\end{aligned}$$

where $\vec{d}_{j-1}^D = (d_{j-1}^x, d_{j-1}^y)^T$ and $\vec{d}_{j+1}^D = (d_{j+1}^x, d_{j+1}^y)^T$ are the neighboring vertices of $\vec{v}_j^i$ in the overlap D.

Case 4: $\vec{v}_j^{\,i} \in D$ *and* $\exists \vec{w} \in W(D) : \ \vec{v}_j^{\,i} \in adj(\vec{w})$
In the last case, the considered vertex is an overlap vertex and also adjacent to at least one intersection point. Thus, we have to add both dynamics from the last two cases and then use the partial derivative for inside vertices.

$$
\begin{aligned}
\nabla_{\vec{v}_j^{\,i}} \frac{1}{k} A(D)^k &= A_D^{k-1} \nabla_{\vec{v}_j^{\,i}} A(D) \\
&= A_D^{k-1} \left(\mathbf{1}_{V(D)}(\vec{v}_j^{\,i}) \nabla_{\vec{v}_j^{\,i}} A(D) + \sum_{\vec{w} \in W(D)} \mathbf{1}_{adj(w)}(\vec{v}_j^{\,i}) D_{\vec{v}_j^{\,i}} \vec{w} \, \nabla_{\vec{w}} A(D) \right) \\
&= A_D^{k-1} \Bigg(\mathbf{1}_{V(D)}(\vec{v}_j^{\,i}) \nabla_{\vec{v}_j^{\,i}} A(D) + \\
&\quad + \sum_{\vec{w} \in W(D)} \left(\mathbf{1}_{out(w)}(\vec{v}_j^{\,i}) D_{\vec{v}_{out}^{\,i}} \vec{w} \, \nabla_{\vec{w}} A(D) + \mathbf{1}_{in(w)}(\vec{v}_j^{\,i}) D_{\vec{v}_{in}^{\,i}} \vec{w} \, \nabla_{\vec{w}} A(D) \right) \Bigg) \\
&= A_D^{k-1} \Bigg(\mathbf{1}_{V(D)}(\vec{v}_j^{\,i}) \nabla_{\vec{v}_j^{\,i}} A(D) + \\
&\quad + \sum_{\vec{w} \in W(D)} \left(\mathbf{1}_{out(w)}(\vec{v}_j^{\,i}) D_{\vec{v}_{out}^{\,i}} \vec{w} + \mathbf{1}_{in(w)}(\vec{v}_j^{\,i}) D_{\vec{v}_{in}^{\,i}} \vec{w} \right) \nabla_{\vec{w}} A(D) \Bigg),
\end{aligned}
$$

where $out(w)$, $in(w) \subset adj(w)$ *denote the sets of outside and inside overlap-adjacent vertices, respectively.*
The difference between $\nabla_{\vec{v}_j^{\,i}} A(D)$ *and* $\nabla_{\vec{w}} A(D)$ *is, that* $\nabla_{\vec{v}_j^{\,i}} A(D)$ *uses the neighboring overlap vertices of* $\vec{v}_j^{\,i}$ *itself in the Area Gradient Formula 6, whereas* $\nabla_{\vec{w}} A(D)$ *uses the neighbors of the according intersection point in* D *which is not the same overlap vertex as* $\vec{v}_j^{\,i}$.
Overall:
Actually, Case 4 already provides the final formulation, since in the other cases the additional terms vanish due to the indicator functions.

□

Figure **??** illustrates the interaction between two overlapping cells, highlighting the effect of the overlap force on their vertices.

Bounce overlap force

While the previously introduced overlap force effectively reduces cell overlap by deforming the cells' shapes, it does not directly separate them spatially—leaving cells temporarily stuck together, relying on random Brownian motion to diffuse apart. With just that force, it is hard to compare the DF cell model to the hard sphere cell model, where overlaps are solved by reflecting them away from each other, resulting in a real distance that both cells have after a really small amount of time.
To address this limitation and ensure a smoother conceptual and mechanical transition from the hard disc cell model, where non deformable cells simply bounce off one

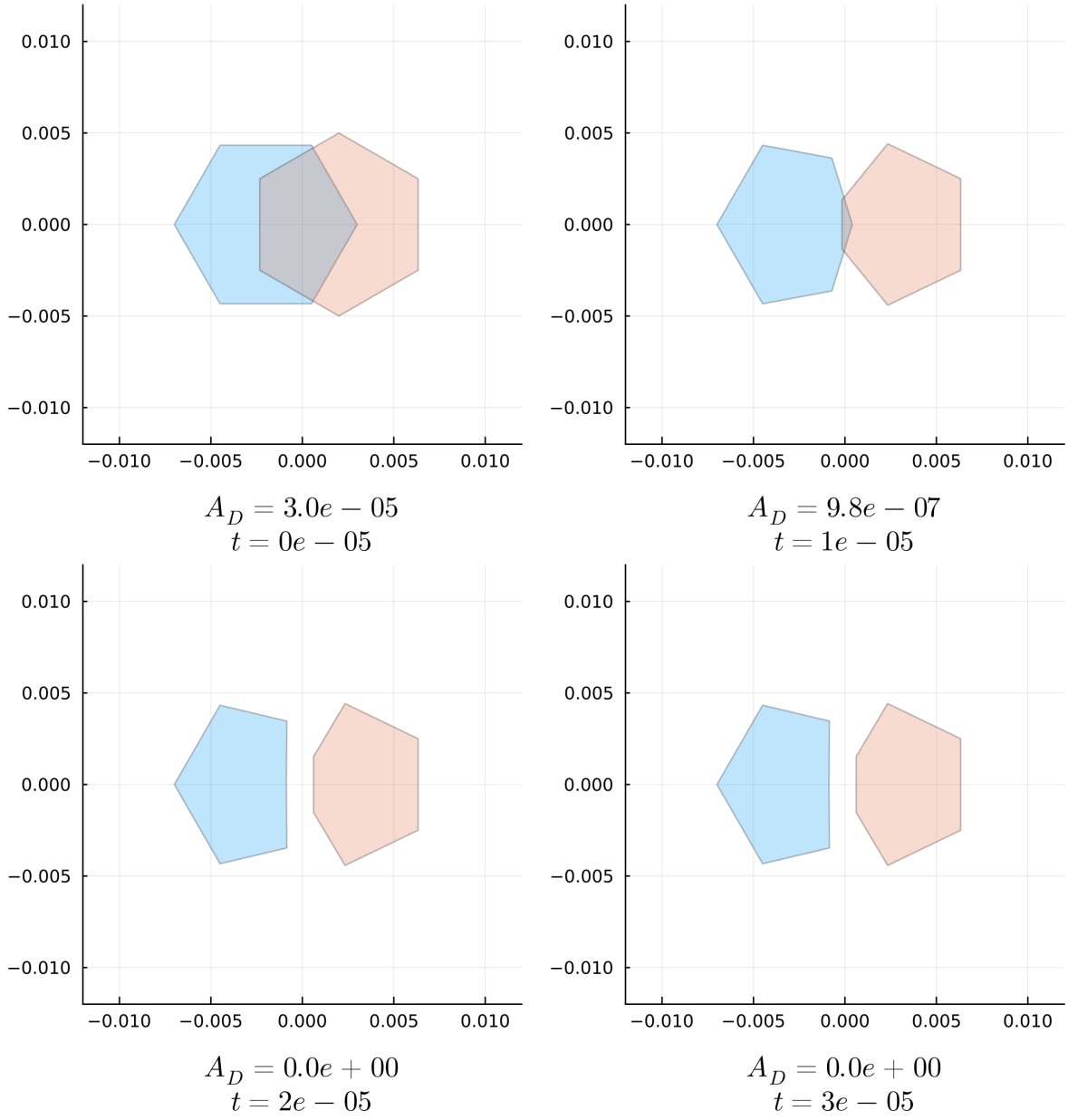

Figure 15: The top four plots show the evolution of a DF cell influenced solely by the deforming overlap force, with $k = 1$ applied to the vertices and a force scaling of 6×10^4, at times $t \in \{0, 1 \times 10^{-5}, 2 \times 10^{-5}, 3 \times 10^{-5}\}$.
In this case, we have $\frac{\mathrm{d}\vec{v}}{\mathrm{d}t} = -6 \times 10^4 \nabla_{\vec{v}} \hat{O}_1(C_1, C_2)$ for all vertices. The overlap area D is indicated below each plot. Click *here* to view the corresponding animation (GIF). Initially, the overlap area is 3×10^{-5}, which is relatively large compared to the cell area of 6.5×10^{-5}. However, it is completely resolved after just two time steps, as also illustrated in the energy diagram in Figure 16.

another, we introduce a second overlap degeneration force. This additional force, which we refer to as the bounce overlap force, acts not by changing cell shape but by actively transporting overlapping cells away from each other. This mechanism captures the spatial repulsion characteristic of rigid body interactions while comple-

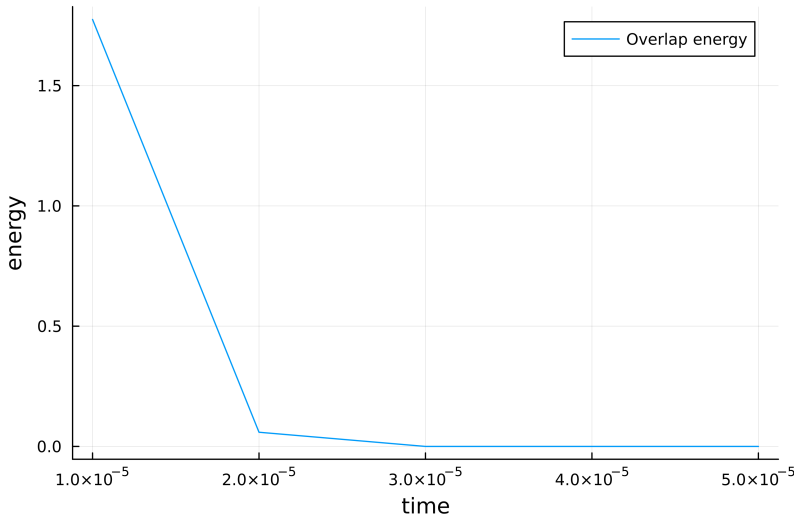

Figure 16: The deforming overlap force resolves the overlap in two time steps.

menting the shape based degeneration of overlaps in deformable cells. In the end, we will use a combination of both overlap forces to get a nice transition from the HCSM to the DF cell model.
In [BC12] overlapping cells with a radius of r that have a centre-to-centre distance of $2r - a$ will be reflective apart in one time step (that is 10^{-5}), resulting in a distance of $2r + a$ between the two cell centres afterwards.
The following force does the exact same for our DF cells. But we need the following assumptions:

1. Our DF cells model circular discs.
2. The cells have a radius of $r \in \mathbb{R}_{>0}$.
3. We can compute the cell centre $\vec{x} = \frac{1}{N_V} \sum_{j=1}^{N_v} \vec{v}_j$ which will be used to determine the distance between two cells.

Definition 4.16. Bounce overlap force
Let us consider two DF cells C_i and C_l, with centres at $\vec{x}_i$ and $\vec{x}_l$, respectively. We assume that each cell has a fixed radius $r > 0$. We define the vector $\mathrm{d}\bar{o}_{i,l} \in \mathbb{R}^2$, which is applied equally to all vertices of cell C_i, representing the repulsive overlap force caused by cell C_l. It is given by

$$\mathrm{d}\bar{o}_{i,l} = \mathbf{1}_{\|\vec{x}_i - \vec{x}_l\|_2 < 2r}(2r - \|\vec{x}_i - \vec{x}_l\|_2)\frac{\vec{x}_i - \vec{x}_l}{\|\vec{x}_i - \vec{x}_l\|_2}.$$

This force is zero if the distance between the cell centres satisfies $\|\vec{x}_i - \vec{x}_l\|_2 \geqslant 2r$. Otherwise, if the cells overlap (i.e., $\|\vec{x}_i - \vec{x}_l\|_2 < 2r$), the vector $\mathrm{d}\bar{o}_{i,l}$ points from $\vec{x}_l$ to $\vec{x}_i$ and has magnitude equal to the overlap depth $a = 2r - \|\vec{x}_i - \vec{x}_l\|_2$. At the same time in the simulation, the same magnitude of displacement is applied to all vertices of C_l in the opposite direction, i.e., along $\vec{x}_l - \vec{x}_i$. This means that all vertices of

cell C_i are displaced away from C_l in the direction $\vec{x}_i - \vec{x}_l$ such that the resulting displacement is sufficient to separate the two cells' centres by exactly $2r + a$.
The force $F_{i,l}^{(\bar{O})}$ that acts on cell C_i due to its interaction with cell C_l is given by

$$F_{i,l}^{(\bar{O})}(C_i, C_l) = (\mathrm{d}\bar{o}_{i,l}, \ldots, \mathrm{d}\bar{o}_{i,l})^T \in \mathbb{R}^{2N_V},$$

where the vector $\mathrm{d}\bar{o}_{i,l}$ is repeated N_V times, once for each vertex of cell C_i.
The total bounce overlap force $F_i^{(\bar{O})} : (\mathbb{R}^{2N_V})^{N_C} \to \mathbb{R}^{2N_V}$ acting on cell C_i due to all other cells in the system is then defined as

$$F_i^{(\bar{O})}(\vec{C}) = \sum_{l \neq i} F_{i,l}^{(\bar{O})}(C_i, C_l).$$

In order to achieve a similarly fast degeneration of the overlap as in [BC12], we need to scale the force with the scaling factor $\alpha^{(\bar{O})} = 10^5$ as the time needed to resolve such an overlap in [BC12] was always one time step which is 10^{-5}.
To account for varying cell stiffness, we introduce a new parameter $h \in [0, 1]$ that controls how 'hard' the cells are. The total overlap force is then defined as a weighted combination of two overlap force types:

$$F^{(\mathbf{O})} = h \cdot F^{(\bar{O})} + (1 - h) \cdot F^{(\hat{O})},$$

where $F^{(\bar{O})}$ denotes the bounce-off overlap force and $F^{(\hat{O})}$ the shape-deforming overlap force.
When $h = 1$, the cells are maximally stiff. In this case, shape deformation is entirely suppressed: all overlaps are resolved solely through the bounce off mechanism, and the shape preserving forces become redundant since the cells always retain their desired configurations.
As h decreases, the cells become progressively softer. The influence of the bounce-off force diminishes, while the shape-deforming overlap force gains dominance, allowing cells to deform more in response to contact with neighbors.
With the introduction of the hardness parameter h, we have established a mechanism that enables a smooth transition from the HSCM dynamics introduced in [BC12] to our new DF cell model.
For $h = 1$, the dynamics are identical to the original HSCM model, as we will show in the following chapter. By gradually decreasing h, we can continuously adapt the system behavior toward the pure deformable (DF) cell model. In this way, we can systematically investigate how the dynamics evolve between the two regimes. In the limiting case $h = 0$, we recover the DF dynamics without the bounce overlap force term.

4.5 The DF SDE

Having defined all individual force contributions acting on the cell vertices, we now combine them to formulate the full dynamics of the system in terms of a stochastic differential equation.
One challenging aspect that remains is the appropriate scaling of all forces. It has

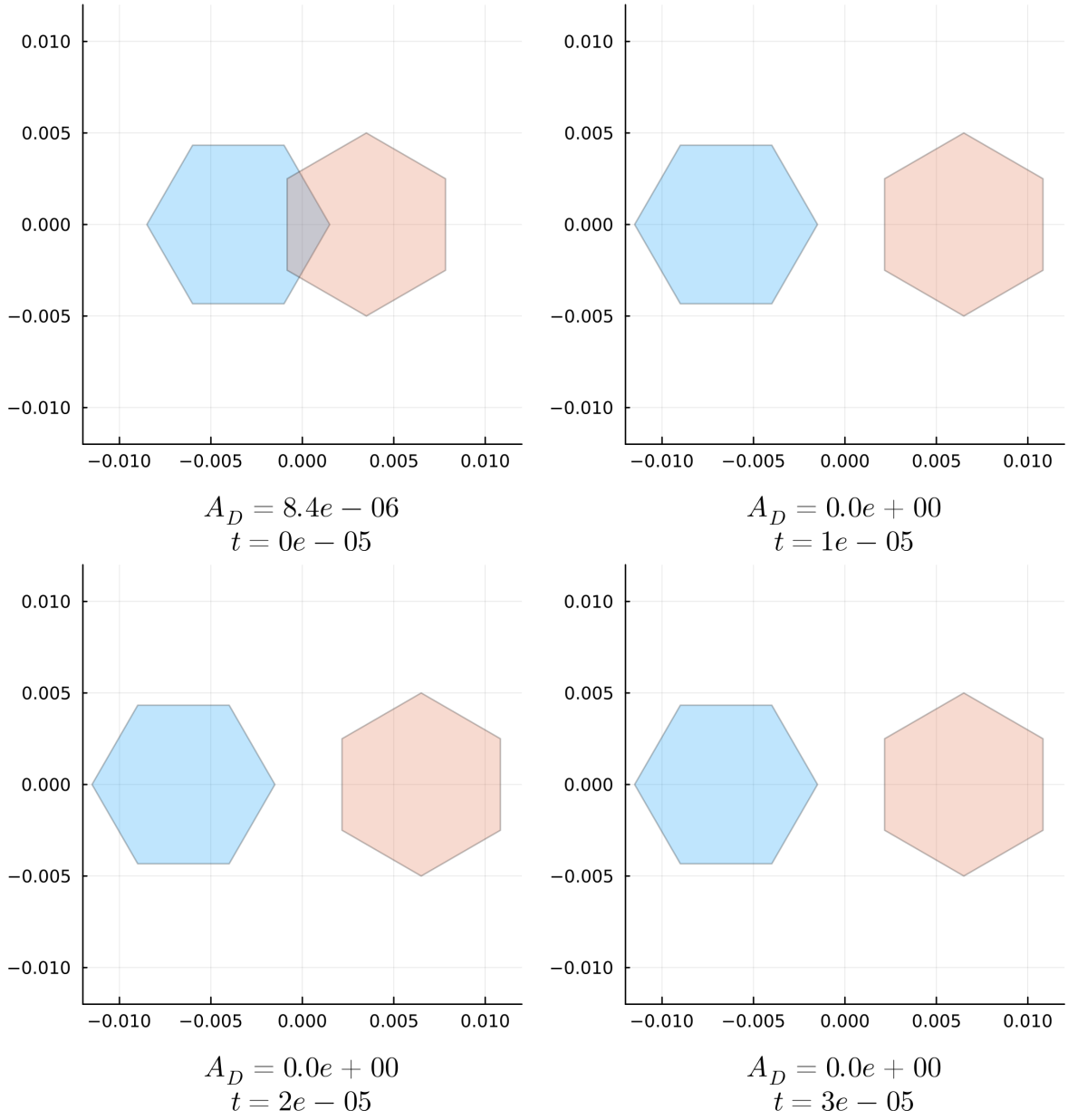

Figure 17: The top four plots present the evolution of a DF cell governed solely by the bounce overlap force, applied to the vertices with a force scaling of 1×10^5, at times $t \in \{0, 1 \times 10^{-5}, 2 \times 10^{-5}, 3 \times 10^{-5}\}$.
In this case, we have $\frac{\mathrm{d}C_i}{\mathrm{d}t} = 1 \times 10^5 \, F_i^{\bar{O}}(C_1, C_2)$ for both cells. Click *here* to view the corresponding animation (GIF).
The overlap vanishes within the very first time step, leaving a visible gap between the two cells. The corresponding energy development is illustrated in Figure 18.

become evident that the system is highly sensitive to these scaling parameters. If the shape-recovering forces are too small, the recovery process is excessively slow. Conversely, if they are too large, the numerical integration scheme tends to become unstable.
To ensure numerical stability while preserving the intended dynamics, we system-

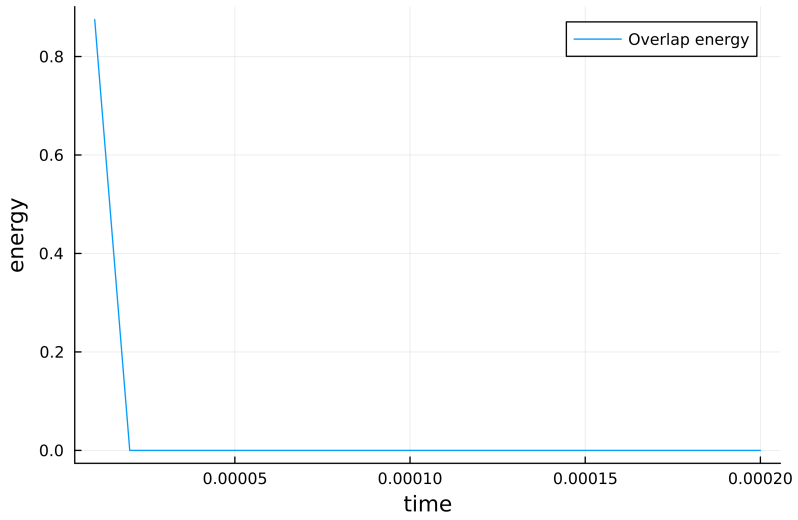

Figure 18: The overlap energy reaches zero after one iteration.

atically tested the appropriate scaling for each force type, focusing in particular on the shape-preserving forces and the deforming overlap force.
Our method was to isolate each of these forces in simulation and determine the threshold at which the system becomes unstable. Specifically, we ran simulations with only one force active at a time and gradually increased its scaling factor until numerical instabilities emerged. We then selected the **maximum stable scaling** as the operative value for that force.
These tests were conducted using a fixed time step of 10^{-5}, with cell configurations composed of six vertices. To rigorously challenge the model, we initialized the cells in deliberately distorted and uncomfortable shapes, which are most prone to triggering instabilities. The configurations used in these tests are the same as those shown in the figures throughout this chapter, where the isolated effects of each force were illustrated following their respective introductions. This allowed us to verify that the chosen scalings are robust even under unfavorable conditions.
For the bounce overlap force, a scaling factor of 10^5 is necessary to reproduce the original bounce off dynamics used in [BC12].
Summarizing all these efforts, we arrived at the following force scalings used in the simulations:

Force type	**Scaling parameter**
Area force	$\alpha_A = 4.0 \times 10^8$
Edge force	$\alpha_E = 3.0 \times 10^4$
Interior angle force	$\alpha_I = 1.0 \times 10^{-1}$
Deforming overlap force	$\alpha_{\hat{O}} = 6.0 \times 10^4$
Bounce overlap force	$\alpha_{\bar{O}} = 1.0 \times 10^5$

Table 2: Scaling parameters for different force types

These scaling par ameters serve as the foundation for the complete DF SDE model, which we now introduce.

Definition 4.17. The DF SDE
We define the vectors

$$e^x_{N_V} = (1,0,1,0,\ldots,1,0)^T, \quad e^y_{N_V} = (0,1,0,1,\ldots,0,1)^T \in \mathbb{R}^{2N_V},$$

which allow us to distribute a two dimensional Brownian motion

$$\mathrm{d}\vec{B}^i_t = (\mathrm{d}B^{i,x}_t, \mathrm{d}B^{i,y}_t)^T$$

to the x and y coordinates of a cell's vertices, respectively. The cell hardness parameter $h \in [0,1]$ is assumed to be given. The deterministic part $F^i(\vec{C})$ of the **DF SDE** is given by

$$\mathbf{F}^i : (\mathbb{R}^{2N_V})^{N_C} \to \mathbb{R}^{2N_V}$$

$$\begin{aligned} \mathbf{F}^i(\vec{C}) = \alpha_A F^{(A)}_2(C_i) + \alpha_E F^{(E)}_2(C_i) + \alpha_I F^{(I)}_2(C_i) + \\ (1-h)\alpha_{\hat{O}} F^{(\hat{O})}_{1,i}(\vec{C}) + h\alpha_{\bar{O}} F^{(\bar{O})}_i(\vec{C}) \end{aligned} \tag{15}$$

for each cell $1 \leqslant i \leqslant N_C$.
Each scaling parameter $\alpha \geqslant 0$. Each F represents one of our forces that got defined in this chapter. For the shape preserving forces, we choose $k = 2$ as then the difference between current state and desired state influences the intensity of the force which is nice. But for the deforming overlap force, we choose $k = 1$ we want it to have a strong impact whenever a small overlap arises. This configuration seems to be the best to model the physics we would like to achieve. With that we can write down the cell wise formulated DF SDE:

$$\mathrm{d}C_i = \mathbf{F}^i(\vec{C})\mathrm{d}t + \mathrm{d}B^{i,x}_t\, e^x_{N_V} + \mathrm{d}B^{i,y}_t\, e^y_{N_V}.$$

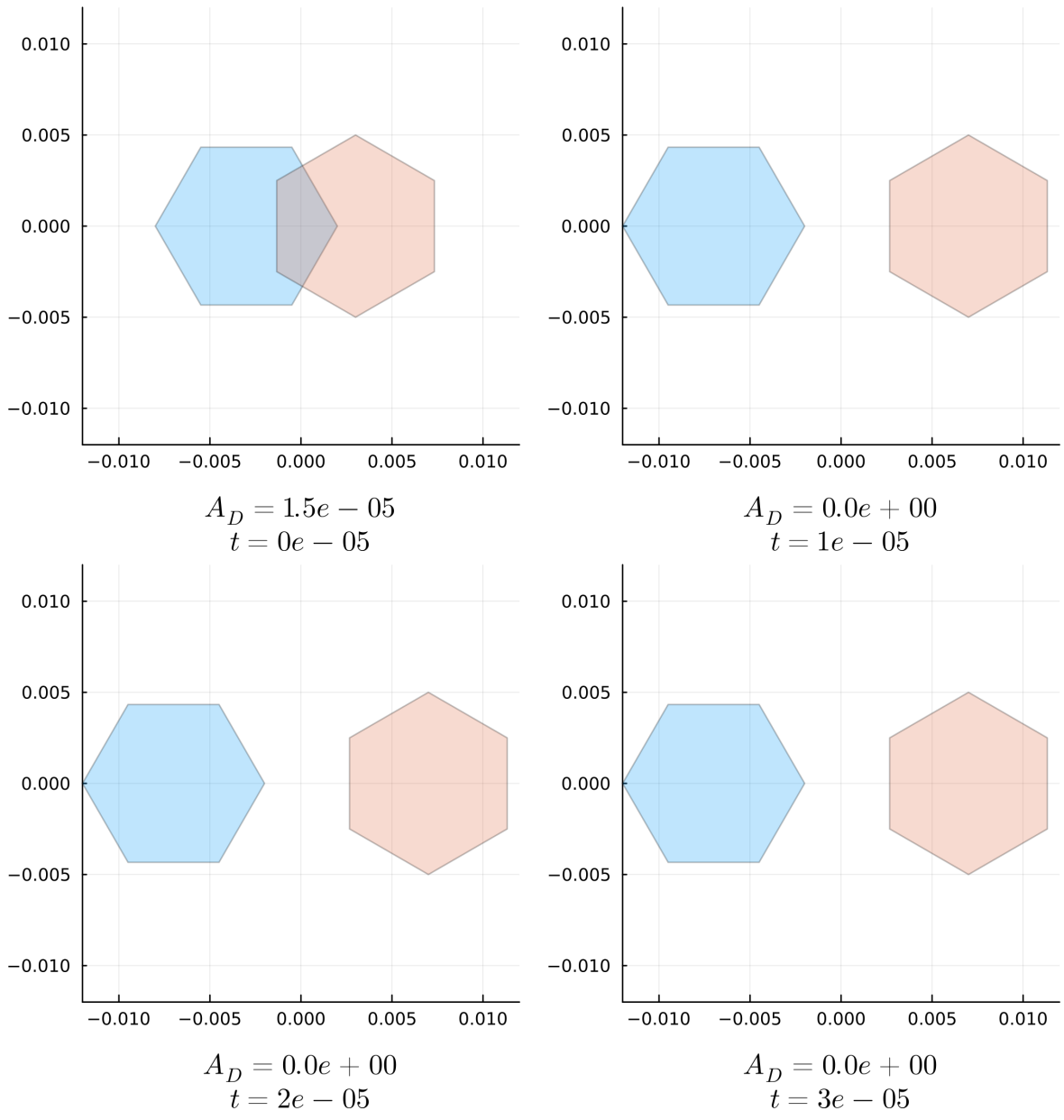

Figure 19: This simulation shows two DF cells evolving according to the dynamics for $i = 1, 2$ $\frac{\mathrm{d}C_i}{\mathrm{d}t} = \mathbf{F}^i(C_1, C_2)$ with hardness $h = 1$, force scalings as listed in Table 2 and without Brownian motion. Click *here* to view the corresponding animation (GIF). The cells are initially generated with overlap. Then, the dynamics from $\mathbf{F}^i$ alone resolve the overlap. Since hardness $h = 1$ was chosen, no deforming overlap force is active and the cell shape remains unchanged. Consequently, the shape-preserving forces are inactive, as the cells stay in their desired states.
This is also reflected in the energy diagram in Figure 20.

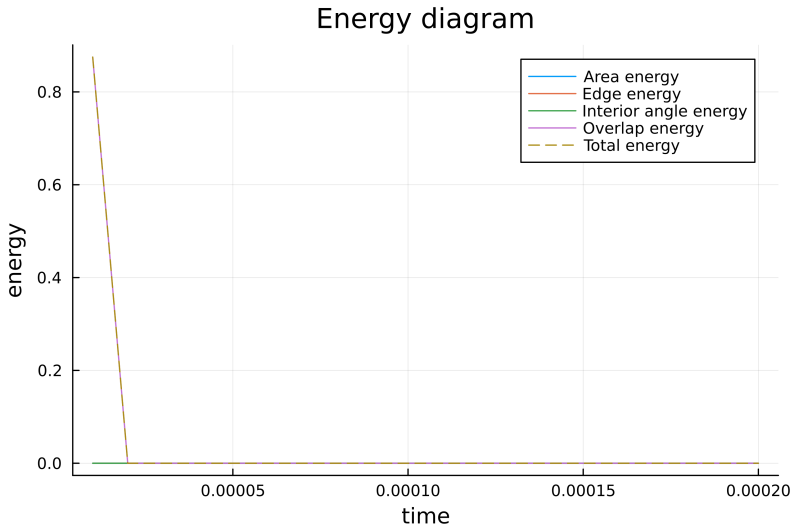

Figure 20: The bounce overlap force eliminates the overlap within a single time step. The energy diagram shows that the overlap energy drops to zero immediately, while the other energies remain constant at zero as the cell shapes do not change.

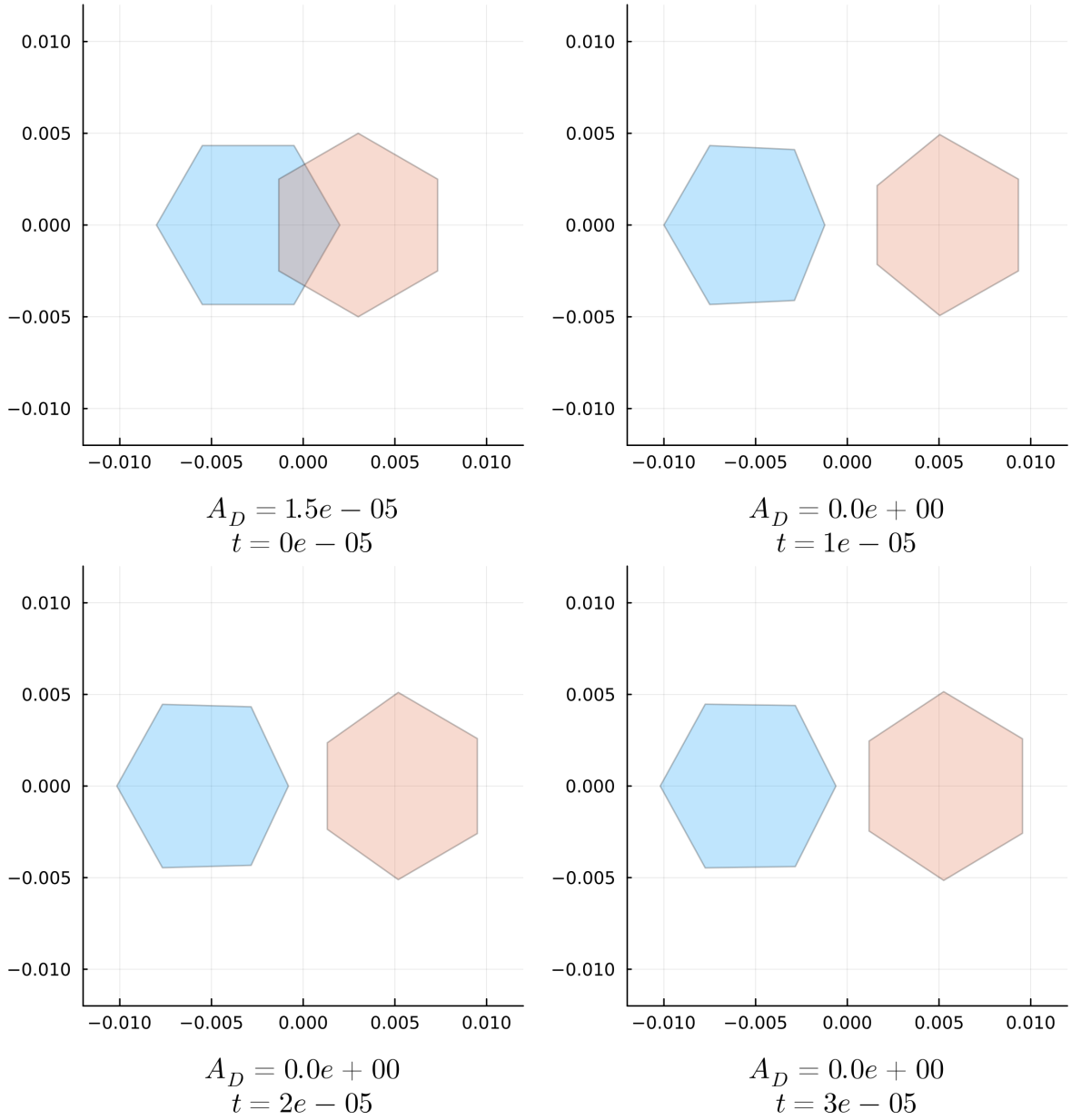

Figure 21: This simulation shows two DF cells again evolving according to the dynamics $i = 1, 2$ $\frac{\mathrm{d}C_i}{\mathrm{d}t} = \mathbf{F}^i(C_1, C_2)$, with hardness $h = 0.5$, force scalings as listed in Table 2, and without Brownian motion. Click *here* to view the corresponding animation (GIF). The cells are initially generated with overlap. Afterwards, the dynamics from $\mathbf{F}^i$ alone resolve the overlap.
In contrast to the previous simulation, the cell shapes now change because the deforming overlap force is active. The overlap is still removed within a single time step. By time step 10, the cell shapes are nearly restored, as also illustrated in Figure 22.

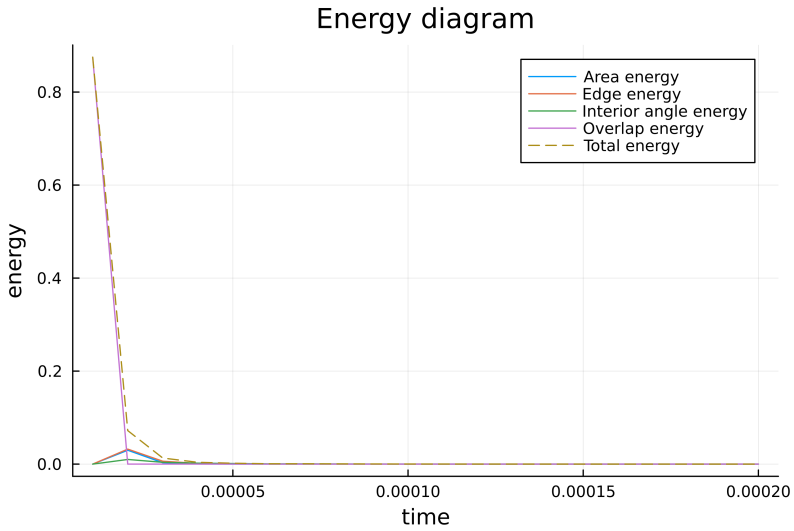

Figure 22: The energy diagram shows that the overlap energy drops to zero immediately, while the shape preserving energies increase initially as the cells deform to resolve the overlap. By time step 10, the cell shapes are nearly restored, which is reflected in the decrease of these energies.

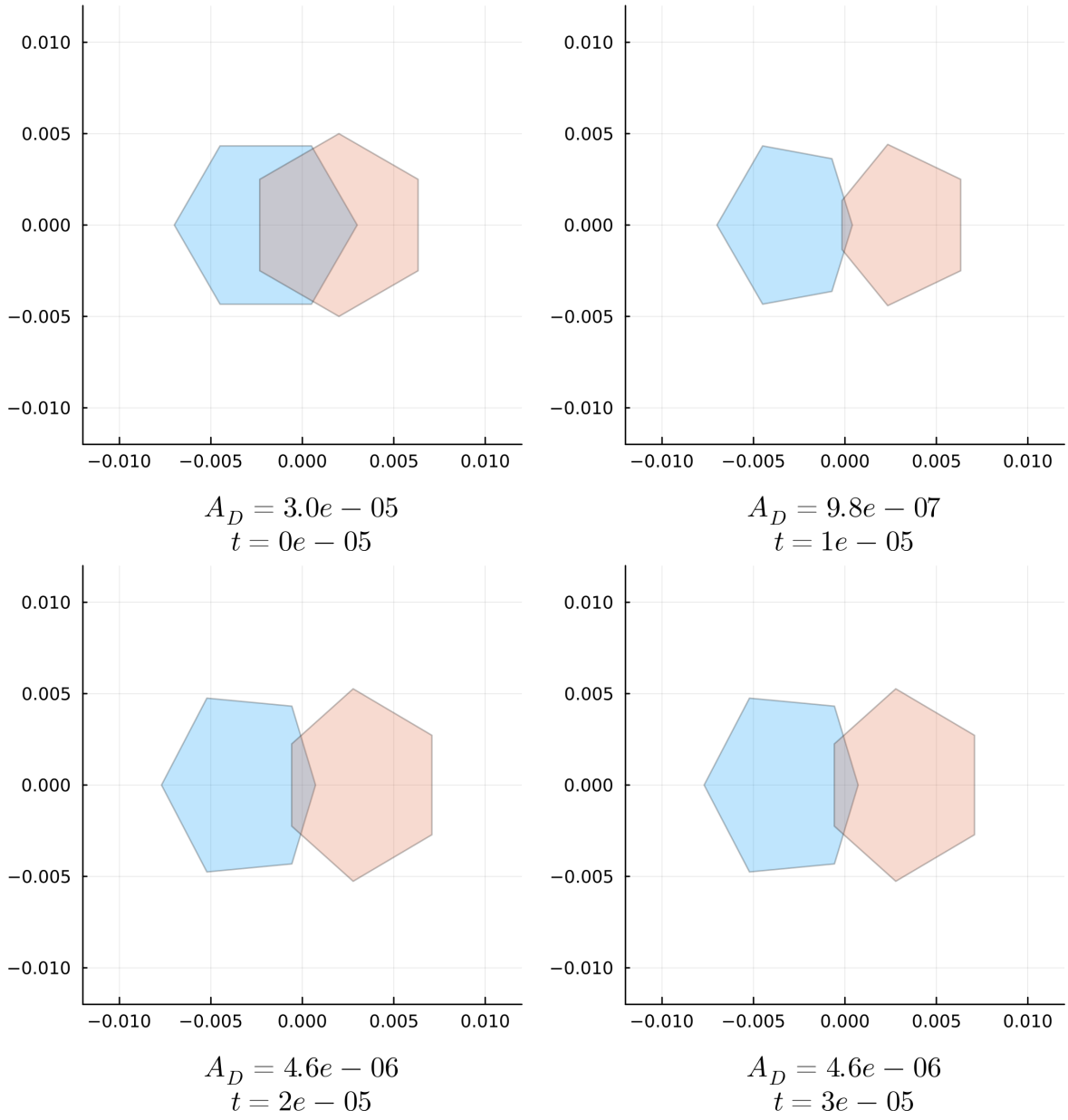

Figure 23: This simulation shows two DF cells evolving according to the dynamics for $i = 1, 2$ $\frac{\mathrm{d}C_i}{\mathrm{d}t} = \mathbf{F}^i(C_1, C_2)$, with hardness $h = 0$, force scalings as listed in Table 2 and without Brownian motion. Click *here* to view the corresponding animation (GIF). The cells are initially generated with overlap. Afterwards, the dynamics from $\mathbf{F}^i$ alone attempt to resolve the overlap.
In this case, only the deforming overlap force is active. This leads to a repeating interplay: the overlap is reduced, the cell shape is restored, and this restoration again induces overlap. Under this setup, neither the overlap nor the desired cell shapes are fully resolved within 20 time steps, although all energy levels remain comparatively low. We can also see this in the energy diagram in Figure 24.

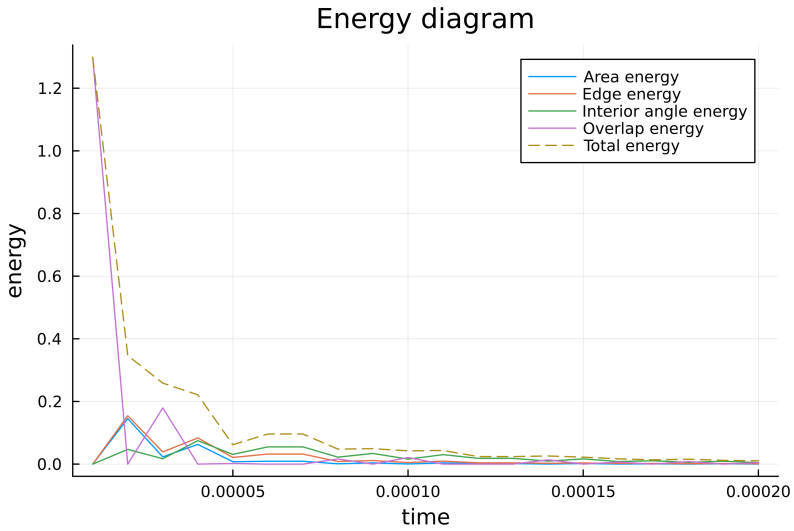

Figure 24: The energy diagram shows that the overlap energy drops initially but then increases again as the cells deform to restore their shapes. This deformation again induces overlap, leading to a repeating cycle. But overall, the total energy converges to a low level, indicating that the system stabilizes over time.

5 Sanity check

Having introduced our cell dynamics, we now want to take a look at the simulation results. Therefore, we aim to compare our simulation results to results from an established cell model from [BC12]. In [BC12] the diffusion dynamics of first a point particle model and second a hard sphere model is studied. Thereby, the two density distributions:

- the joint probability density function $P(\vec{X}, t)$ of the system of all cell centres $\vec{X}$ at time t,
- the marginal distribution function of the first particle $p(\vec{x}_1, t)$

play an important roll.
The joint probability density function $P(\vec{X}, t)$ is a function describing the positions of all particles in the system, while the marginal distribution function $p(\vec{x}_1, t)$ is a function describing only the position of the first particle.
It is sufficient to consider only the marginal distribution function of first particle, because all particle act similarly.
Gaining $p(\vec{x}_1, t)$ from $P(\vec{X}, t)$ is a big reduction of complexity, since we reduce from a high-dimensional PDE for P to a low-dimensional PDE for p. The marginal distribution function the of first particle can always be determined via

$$p(\vec{x}_1, t) = \int P(\vec{X}, t) \mathrm{d}\vec{x}_2 \dots \mathrm{d}\vec{x}_{N_{BC}}.$$

5.1 Reference simulations: Bruna and Chapman (2012)

The most simple model that gets considered for the diffusion dynamics of cell systems is the point particle model. Here the cells get modeled with sizeless points that perform a Brownian motion on the domain.
Since the cells do not have a real size, no interaction between the cells can occur, since they will never hit upon each other.
The paper [BC12] analyses these dynamics on the domain

$$\Omega_{BC} = [-0.5, 0.5]^2,$$

on which $N_{BC} = 400$ particles are located.
The movement of each point particle $\vec{x}_i$ in the simulation is given by the SDE

$$\mathrm{d}\vec{x}_i(t) = \sqrt{2}\mathrm{d}B_t^{(i)}, \quad 1 \leqslant i \leqslant N_{BC},$$

which describes a Brownian motion in Ω_{BC}. The reflective boundary condition on $\partial\Omega_{BC}$ is imposed. It is known, that the joint probability density of the particle system in this setup evolves according to the diffusion equation, i.e.

$$\frac{\partial P}{\partial t}(\vec{X}, t) = \Delta_{\vec{X}} P = \nabla_{\vec{X}} \cdot [\nabla_{\vec{X}} P] \tag{16}$$

inside of the domain.
Since all particles are independent, we can compute

$$P(\vec{X}) = \prod_{i=1}^{N_{BC}} p(\vec{x}_i, t). \tag{17}$$

Therefore, we obtain the marginal distribution function as

$$\frac{\partial p}{\partial t}(\vec{x}_1, t) = \Delta_{\vec{x}_1} p = \nabla_{\vec{x}_1} \cdot [\nabla_{\vec{x}_1} p]. \tag{18}$$

A next step that results in the hard sphere cell model (HSCM) is to give the cell particles a real size.
Let $0 < \epsilon \ll 1$ be the diameter of all cells that are now two dimensional discs with the same size. This changes the dynamics of the cells immense, since they now have chance to collide into each other which is a form of interaction.
The authors of [BC12] also did a simulation with the HSCM. The setting is as similar as possible to the point particle model, because a main goal of the paper was to compare the diffusion characteristics of both models. There are still $N_{BC} = 400$ cells located on the domain.

The initial condition of both models follows a two dimensional normal distribution with the addition that the distance of each cell centre to all others is at least ϵ. The used distribution $\mathcal{N}_2\left(\begin{pmatrix}0\\0\end{pmatrix}, \begin{pmatrix}0.09^2 & 0\\ 0 & 0.09^2\end{pmatrix}\right)$ has an integral of one over Ω_{BC}.
We can compute this initial condition with Algorithm 5.1.

Algorithm 5.1. ***Computation of the initial cell system***

1. *Generate a point* $\vec{x} \sim \mathcal{N}_2\left(\begin{pmatrix}0\\0\end{pmatrix}, \begin{pmatrix}0.09^2 & 0\\ 0 & 0.09^2\end{pmatrix}\right)$.
2. *If for all already generated centres* $\vec{x}_j : \|\vec{x} - \vec{x}_j\|_2 > \epsilon$ *is true, use* $\vec{x}$ *as the next cell centre, otherwise discard the point and restart with step 1 until* N_{BC} *cell centres are found.*

Since we do not want any overlap to occur during the whole simulation with the HSCM, the feasible domain for the whole cell system is not directly $\Omega_{BC}^{N_{BC}}$, but instead

$$\Omega_{BC}^{\epsilon} = \Omega_1^{\epsilon} \times \ldots \times \Omega_{N_{BC}}^{\epsilon},$$
$$\Omega_i^{\epsilon} = \Omega_{BC} \backslash (\cup_{j \neq i} \mathrm{B}_{\epsilon}(\vec{x}_j)), \quad 1 \leqslant i \leqslant N_{BC},$$

where $\mathrm{B}_{\epsilon}(\vec{x}_j)$ denotes the ball around $\vec{x}_j$ with radius ϵ.
This domain prevents overlaps between the cells by not allowing each cell to drift closer than ϵ to any other cell.
HSCM cells perform the same Brownian motion as the point particles.
The next question is, how cell collisions are modelled. Unlike in our DCF model where cell interactions are modelled as forces acting inside of the domain, the cell collisions from the HSCM arise from the reflective boundary condition.
Let us assume that two cells i and j are given such that $\|\vec{x}_i - \vec{x}_j\|_2 = \epsilon$ is true. Then, both cell centres are located at the boundary $\partial\Omega_{BC}^{\epsilon}$. Here, the reflective boundary condition is still imposed and it causes both cells to bounce of from each other in the direction of the outward normal vector from the excluded area of the respectively other cell.

In [BC12] the authors managed to compute the marginal distribution function of the first particle of the HSCM. In two dimensions it is given by:

$$\frac{\partial p}{\partial t}(\vec{x}_1, t) = \nabla_{\vec{x}_1} \cdot \{\nabla_{\vec{x}_1}[p + \frac{\pi}{2}(N_{BC} - 1)\epsilon^2 p^2]\}. \tag{19}$$

We can see a connection to Equation 18. Let us define a diffusion coefficient

$$D_\epsilon(p) = 1 + \pi(N_{BC} - 1)\epsilon^2 p$$

that depends on the local partical density p and the cell diameter ϵ. For the point particles, we have $\epsilon = 0$ and $D_\epsilon(p) = 1$. Thus, we can rewrite the first marginal to

$$\frac{\partial p}{\partial t}(\vec{x}_1, t) = \nabla_{\vec{x}_1} \cdot [D_\epsilon(p)\nabla_{\vec{x}_1} p].$$

In the case of the hard spheres, where $\epsilon > 0$, we can compute

$$\begin{aligned}
\frac{\partial p}{\partial t}(\vec{x}_1, t) &= \nabla_{\vec{x}_1} \cdot [D_\epsilon(p)\nabla_{\vec{x}_1} p] \\
&= \nabla_{\vec{x}_1} \cdot [(1 + \pi(N_{BC} - 1)\epsilon^2 p)\nabla_{\vec{x}_1} p] \\
&= \nabla_{\vec{x}_1} \cdot [\nabla_{\vec{x}_1} p + \pi(N_{BC} - 1)\epsilon^2 p \nabla_{\vec{x}_1} p] \\
&= \nabla_{\vec{x}_1} \cdot [\nabla_{\vec{x}_1} p + \pi(N_{BC} - 1)\epsilon^2 \frac{1}{2}\nabla_{\vec{x}_1} p^2] \\
&= \nabla_{\vec{x}_1} \cdot [\nabla_{\vec{x}_1}(p + \frac{\pi}{2}(N_{BC} - 1)\epsilon^2 p^2)]
\end{aligned}$$

to recover Equation 19.
When considering $D_\epsilon(p) = 1 + \pi(N_{BC} - 1)\epsilon^2 p$ to be the diffusion coefficient, we can conclude that an increase in the number of cells N_{BC}, the cell diameter ϵ, or the local density p leads to an increased diffusion rate of the system. Overall, we conclude that the bounce effect of the HSCM enhances the diffusion rate of the system's density.
Another evidence of this behavior is shown in Figure 2 in [BC12].
Here, we can see two Monte Carlo simulations. A Monte Carlo simulation is a computational technique that uses random sampling to model and analyse complex systems or processes that are difficult to solve analytically. It repeatedly generates random inputs according to specified probability distributions and computes the resulting outcomes to estimate quantities like averages, variances, or distributions.
In our case, the Monte Carlo simulations are used to track the positions of cell centres over time. Each simulation begins from an initial configuration of cells, which is consistently generated using Algorithm 5.1. After initialization, the prescribed dynamics - either the point particle model or the hard sphere model - are applied, and the positions of the cell centres are recorded at a fixed time point, $t = 0.05$.
To visualise the results, we construct heatmaps representing the spatial distribution of cells at the final time. This is done by discretizing the domain into a uniform grid of sub squares. For each sub square, we count how many cells fall within it across all simulations. The resulting counts are normalised by dividing by the total

number of cells N_C, the number of simulations, and the area of a sub square. This normalisation ensures that the heatmap represents a probability density, satisfying the mass conservation condition:

$$\sum_{i \in \text{ sub squares}} \text{value}_i \cdot \text{area}_i = 1.$$

This approach provides a smooth estimate of the empirical cell density, allowing direct comparison with the corresponding solutions of the diffusion equations.
Figure 3 shows the discussed graphic from [BC12].

5.2 Reproduction of reference results

Before running our new dynamics that include cell flexibility, we first want to guarantee that the simulations are running in the correct setup. Therefore, we started with recreating the Monte Carlo simulation for the point particles. I always fixed the color scale to be the same as in [BC12] in order to gain comparability. The simulation parameters are the same as in [BC12].
All of our simulations run in the Julia programming language. There, we used the package 'DifferentialEquations.jl' with its structure 'SDEProblem()' and then solved it with the package inbuilt Euler Maruyama scheme that uses a constant time step size.
I employed a callback function that was triggered after each simulation step to implement a reflective boundary condition. Whenever a particle moved outside the domain, it was relocated to the position within the domain such that its distance to the domain boundary remained unchanged, effectively reflecting the particle off the boundary.
Beside of this, all particles moved according to the two dimensional Brownian motion

$$\mathrm{d}\vec{x}_i(t) = \sqrt{2}\mathrm{d}\vec{B}^{\,i}, \quad 1 \leqslant i \leqslant N_C.$$

Figure **??** shows the evolution of the particle density in terms of heatmaps for different time steps. The results of our Monte Carlo simulation appear to be in good agreement with those of Bruna and Chapman, suggesting that our approach is robust and accurate.
Next, we consider the HSCM and run the Monte Carlo simulation for a cell diameter of $\epsilon = 0.01$. Figure **??** shows the density evolution of the HSCM.

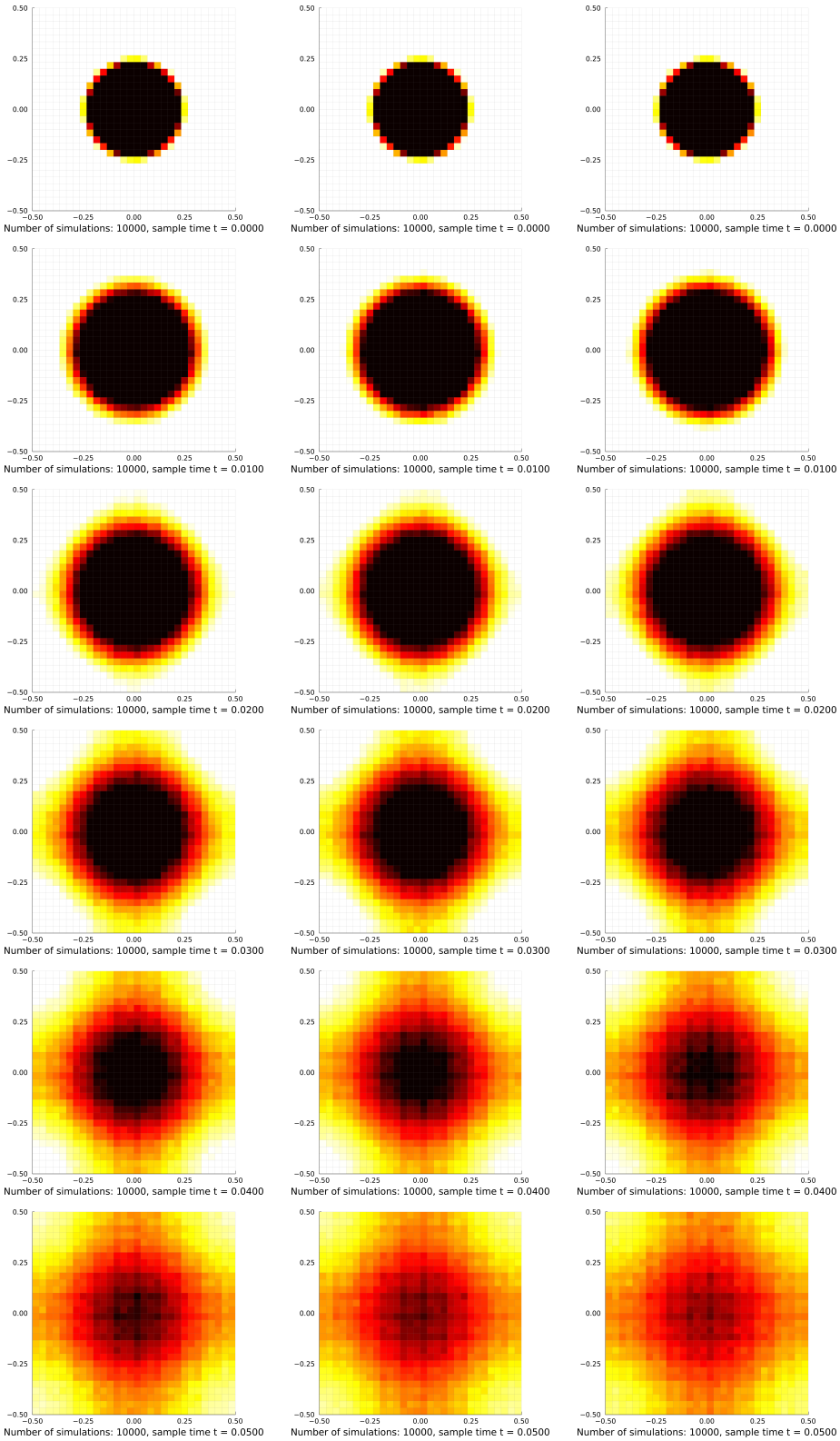

Figure 25: Heatmaps of a Monte Carlo simulation of the DF cell model with different hardness values at the times $t \in \{0.00, 0.01, 0.02, 0.03, 0.04, 0.05\}$. Left column shows hardness 0, we can see hardness 0.5 in the middle and hardness 1 on the right. We can oberserve that the diffusion rate increases with increasing hardness.

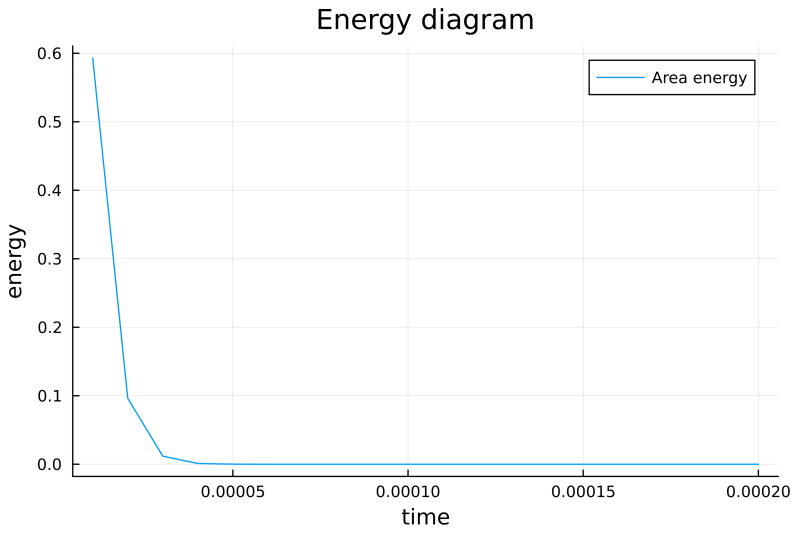

Figure 26: The area force successfully restores the desired cell area after 5 time steps as this energy diagram shows.

6 Density computations

In the previous chapter, we studied Monte Carlo simulations of our DF model and visualised the resulting particle distributions as heatmaps at different time points. These simulations provided valuable insight into the statistical behavior of the system for finite numbers of cells.
In this chapter, we pursue a different approach: instead of extracting approximate density fields from stochastic simulations, we aim to compute the underlying density distribution function ρ that describes the limiting behavior of the system as the number of cells $N_C \to \infty$. More precisely, we seek the density ρ such that the limiting measure μ satisfies $\mathrm{d}\mu = \rho(x)\mathrm{d}x$. Of course, this density will depent on the desired cell state, i.e. the number of vertices and its shape, as well as on the forces we apply on the vertices as the dynamic and their scalings.
As a starting point for this analytical treatment, we consider the empirical measure μ^{N_C}, which encodes the particle configuration at finite N_C, and study its convergence to the continuous measure μ.
The empirical measure $\mu^{N_C} \in \mathcal{P}(\mathbb{R}^2)$ is the starting point of this computation.
Let $\{\vec{x}_i\}_{i=1}^{N_C} \subset \mathbb{R}^2$ be the set of all cells' centre points. We define μ^{N_C} as:

$$\mu^{N_C} : \mathcal{B}(\mathbb{R}^2) \to [0,1]$$
$$\mu^{N_C}(A) = \frac{1}{N_C}\sum_{i=1}^{N_C} \delta_{\vec{x}_i(t)}(A),$$

where $\mathcal{B}(\mathbb{R}^2)$ is the Borel sigma-algebra on $\mathbb{R}^2$ and $\delta_{\vec{x}_i(t)}$ denotes the Dirac measure:

$$\delta_{\vec{x}_i(t)} : \mathcal{B}(\mathbb{R}^2) \to \{0,1\}$$
$$\delta_{\vec{x}_i(t)}(A) = \begin{cases} 1 & \text{if } \vec{x}_i(t) \in A, \\ 0 & \text{if } \vec{x}_i(t) \notin A. \end{cases}$$

For any test function $\phi \in C_c^\infty(\mathbb{R}^2)$, the Dirac measure satisfies

$$\int_{\mathbb{R}^2} \phi(x)\mathrm{d}\delta_{\vec{x}_i(t)}(x) = \phi(\vec{x}_i(t)).$$

For a set $A \in \mathcal{B}(\mathbb{R}^2)$, $\mu^{N_C}(A)$ is the relative proportion of the N_C particles that are located in A.
For a finite $N_C \in \mathbb{N}$, μ^{N_C} is a discrete measure that only has its mass divided on the exact particle locations. As we increase the number of particles, μ^{N_C} will spread out - having more particle locations to cover with each particle having a lower influence on the result of μ^{N_C} as we divide through N_C. This process is quite simular to the transition from a sum to an according integral:

$$\sum_{i=1}^{N_C} \frac{1}{N_C} f(x_i) \xrightarrow{N\to\infty} \int f(x)\mathrm{d}x,$$

where we can also see a transition from a discrete starting problem, having discrete points x_i, to a continious integral where $x \in (a,b)$. As μ^N is a measure that lives on

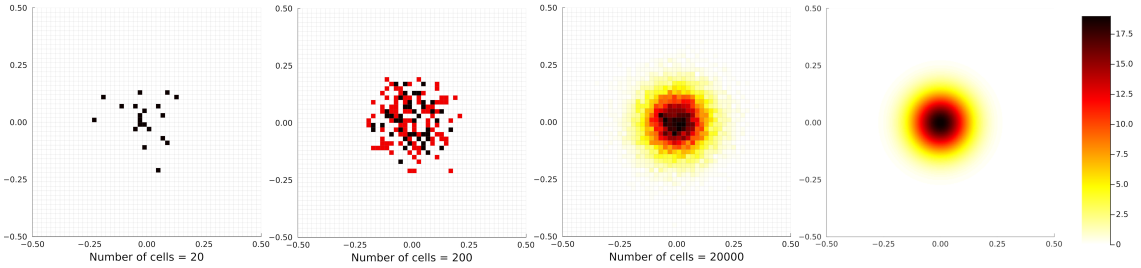

Figure 27: Visualisation of empirical measures μ^{N_C} for increasing numbers of cells, alongside the corresponding theoretical density. This visualisation illustrates the transition $\mu^{N_C} \xrightarrow{N_C \to \infty} \mu$.
All cell configurations are sampled from the same initial condition described in the previous chapter: a two-dimensional normal distribution $\mathcal{N}_2((0,0), \sigma^2 I_2)$ with $\sigma = 0.09$. In the first three subplots, the domain $[-0.5, 0.5]^2$ is discretised into square bins of side length $\frac{1}{50}$. Each bin A corresponds to a measurable set in the definition of the empirical measure $\mu^{N_C}(A) = \frac{1}{N_C} \sum_{i=1}^{N_C} \delta_{x_i}(A)$, where the color intensity encodes the number of cells in that bin.
The first subplot shows a realisation with $N_C = 20$ cells, the second with $N_C = 200$, and the third with $N_C = 20.000$. We observe that as N_C increases, the empirical measure μ^{N_C} becomes a smoother approximation of the underlying density. The color scale is fixed across all subplots to allow direct visual comparison. The fourth panel displays the exact density function of the initial distribution, $\rho(x) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{\|x\|^2}{2\sigma^2}\right)$, with $\sigma = 0.09$.

sets $A \in \mathcal{B}(\mathbb{R}^d)$, we cannot directly plot it as a function. Instead, we try to visualise it meaningfully by using histograms as approximations in Figure 27. This is also a good connection from the previus section, where we also used histograms to show the results of the monte carlo simulations.
The distribution from the fourth subplot in Figure 27 is aimed to be computed for the dynamics of our DF cell model.
In the end, we want to achieve:

$$\mu^{N_C} \xrightarrow{N_C \to \infty} \mu$$

by letting the number of cells go to infinity.

6.1 Transition $\mu^{N_C} \xrightarrow{N_C \to \infty} \mu$

6.2 General energy computation

Lets define our cell centres with:

$$\vec{X} = (\vec{x}_1, \ldots, \vec{x}_N) \in \mathbb{R}^{2N} (\text{vector of all cell centres}),$$
$$\text{for } \vec{x}_i \in \mathbb{R}^2,\ 1 \leqslant i \leqslant N.$$

The energy that gets used for our cell dynamic shall be:

$$E : \mathbb{R}^2 \to \mathbb{R}$$
$$E(\vec{x}_i) = \frac{1}{2}|\vec{x}_i|^2.$$
$$\nabla_{(\vec{x}_i)} E(\vec{x}_i) : \mathbb{R}^2 \to \mathbb{R}^2$$
$$\nabla_{(\vec{x}_i)} E(\vec{x}_i) = |\vec{x}_i|.$$

We define the dynamic of a particle $\vec{x}_i$ via:

$$\frac{\mathrm{d}\vec{x}_i}{\mathrm{d}t} = -\nabla_{\vec{x}_i} E(\vec{x}_i) \in \mathbb{R}^2.$$

We define the probability measure:
(question: is μ defined on a single particle $[\mu^{N_C} \in \mathcal{P}(\mathbb{R}^2)]$ or on the whole particle system $[\mu^{N_C} \in \mathcal{P}(\mathbb{R}^{2N})]$)
(question: what does μ say?
Its 1 when the particle is at a given location? vs Its 1 when the particle system is at a given configuration?)

$$\mu : \mathbb{R}^2 \to [0, \infty)$$

μ is the density of cell system. μ^N is the empirical measure. It takes a subset $A \subset \mathbb{R}^2$ as an argument and gives the relative number of particles that are inside of A.
Let $\phi \in C_c^\infty(\mathbb{R}^2, \mathbb{R})$ (??) be a test function. Its gradient field is $\nabla\phi : \mathbb{R}^2 \to \mathbb{R}^2$. We compute:

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\int \phi \mathrm{d}\mu^N &= \frac{\mathrm{d}}{\mathrm{d}t}(\frac{1}{N_C}\sum_{i=1}^{N_C}\phi(\vec{x}_i)) \\
&= -\frac{1}{N_C}\sum_{i=1}^{N_C}\nabla\phi(\vec{x}_i)\cdot\nabla E(\vec{x}_i) \\
&= -\frac{1}{N_C}\sum_{i=1}^{N_C}\int \nabla\phi(x)\cdot\nabla E(x)\mathrm{d}\delta_{\vec{x}_i} \\
&= -\int \nabla\phi(x)\cdot\nabla E(x)\mathrm{d}\mu^N\mathrm{d}x \\
&= \int \phi(x)\nabla\cdot(\mu^N(\nabla E(x)))\mathrm{d}x
\end{aligned}$$

$$\Rightarrow 0 = \partial_t\rho - \nabla(\rho_v),$$

where ρ is the density function of μ such that

$$\mu(dx) = \rho(x)dx.$$

question: what is the space we integrate above? (i gues $\mathbb{R}^{2N}$)

7 Conclusion

In this work, we develop a vertex-based discrete form (DF) model that captures the behavior of deformable cells with realistic mechanical interactions, including shape preservation, cell-cell overlap resolution, and stochastic motion.

8 Outlook

An interesting extension of the current model would involve assigning individual desired states to each cell, in contrast to the uniform desired state used throughout this study. This modification would naturally lead to cell-specific energies and corresponding forces, as both would depend on the unique desired configuration of each cell. Incorporating such heterogeneity could allow the model to capture more complex biological behaviors, such as differentiation, cell-type-specific migration, or adaptive responses to environmental cues. - curved surfaces - 3d - cell division - more parameter studies - use more vertices - limit $N_V \to \infty$ - overdamping - new shapes - Additionally, many vertex models incorporate rules that govern changes in connection among vertices, and therefore allow for changes in cell neighbor relationships. -¿ These approximations are suitable in the case of tightly packed cell sheets, where the intercellular space is negligible, and is based on experimental observations that cells in epithelial tissues are often arranged in polygonal or polyhedral structures (19) and can move around relative to other cells (20)

Statement of authorship

I hereby declare that I have written this thesis (*Derivation and study of a non-confluent model for deformable cells*) under the supervision of Jun.-Prof. Dr. Markus Schmidtchen independently and have listed all used sources and aids. I am submitting this thesis for the first time as part of an examination. I understand that attempted deceit will result in the failing grade „not sufficient“ (5.0).

Tim Vogel

Dresden, October 15, 2025

Technische Universität Dresden

Matriculation Number: 4930487

References

[AT20] Ricard Alert and Xavier Trepat. Physical models of collective cell migration. *Annual Review of Condensed Matter Physics*, 11(1):77–101, 2020.

[BC12] Maria Bruna and S. Jonathan Chapman. Excluded-volume effects in the diffusion of hard spheres. *Phys. Rev. E*, 85:011103, Jan 2012.

[BCR17] Maria Bruna, S. Jonathan Chapman, and Martin Robinson. Diffusion of particles with short-range interactions. *SIAM Journal on Applied Mathematics*, 77(6):2294–2316, 2017.

[BSY+18] Arman Boromand, Alexandra Signoriello, Fangfu Ye, Corey S. O'Hern, and Mark D. Shattuck. Jamming of deformable polygons. *Phys. Rev. Lett.*, 121:248003, Dec 2018.

[CCP+14] Danfeng Cai, Shann-Ching Chen, Mohit Prasad, Li He, Xiaobo Wang, Valerie Choesmel-Cadamuro, Jessica K Sawyer, Gaudenz Danuser, and Denise J Montell. Mechanical feedback through e-cadherin promotes direction sensing during collective cell migration. *Cell*, 157(5):1146–1159, 2014.

[CV15] Andrew G Clark and Danijela Matic Vignjevic. Modes of cancer cell invasion and the role of the microenvironment. *Curr Opin Cell Biol*, 36(13-22):10–1016, 2015.

[DRSB+05] Olivia Du Roure, Alexandre Saez, Axel Buguin, Robert H Austin, Philippe Chavrier, Pascal Siberzan, and Benoit Ladoux. Force mapping in epithelial cell migration. *Proceedings of the National Academy of Sciences*, 102(7):2390–2395, 2005.

[FG09] Peter Friedl and Darren Gilmour. Collective cell migration in morphogenesis, regeneration and cancer. *Nature reviews Molecular cell biology*, 10(7):445–457, 2009.

[FNW+95] Peter Friedl, Peter B Noble, Paul A Walton, Dale W Laird, Peter J Chauvin, Roger J Tabah, Martin Black, and Kurt S Zänker. Migration of coordinated cell clusters in mesenchymal and epithelial cancer explants in vitro. *Cancer research*, 55(20):4557–4560, 1995.

[FOBS14] Alexander G. Fletcher, Miriam Osterfield, Ruth E. Baker, and Stanislav Y. Shvartsman. Vertex models of epithelial morphogenesis. *Biophysical Journal*, 106(11):2291–2304, 2014.

[GT18] Matthew Good and Xavier Trepat. Cell parts to complex processes, from the bottom up, 2018.

[Her32] Earl H Herrick. Mechanism of movement of epidermis, especially its melanophores, in wound healing, and behavior of skin grafts in frog tadpoles. *The Biological Bulletin*, 63(2):271–286, 1932.

[Hol14] SJ Holmes. The behavior of the epidermis of amphibians when cultivated outside the body. *J. exp. Zool*, 17(2):281–295, 1914.

[HV23] Lea Happel and Axel Voigt. Coordinated motion of epithelial layers on curved surfaces. *Cluster of Excellence, Physics of Life, TU Dresden*, 2023.

[JGS18] Frank Jülicher, Stephan W Grill, and Guillaume Salbreux. Hydrodynamic theory of active matter. *Reports on Progress in Physics*, 81(7):076601, 2018.

[MJR+13] M Cristina Marchetti, Jean-François Joanny, Sriram Ramaswamy, Tanniemola B Liverpool, Jacques Prost, Madan Rao, and R Aditi Simha. Hydrodynamics of soft active matter. *Reviews of modern physics*, 85(3):1143–1189, 2013.

[PJJ15] Jacques Prost, Frank Jülicher, and Jean-François Joanny. Active gel physics. *Nature physics*, 11(2):111–117, 2015.

[RCCT17] Pere Roca-Cusachs, Vito Conte, and Xavier Trepat. Quantifying forces in cell biology. *Nature cell biology*, 19(7):742–751, 2017.

[Sho14] ShoelaceFormula. Green's theorem and area of polygons. blogoverflow, June 2014. Published by: apnorton. URL: https://math.blogoverflow.com/2014/06/04/greens-theorem-and-area-of-polygons/. LAst accessed on 23.11.2023.

[Sho22] ShoelaceIllustration. Deriving the trapezoid formula. URL: https://commons.wikimedia.org/wiki/File:Trapez-formel-prinz.svg, January 2022. Published by user 'Ag2gaeh'. Last accessed on 23.11.2023.

[TWA+09] Xavier Trepat, Michael R Wasserman, Thomas E Angelini, Emil Millet, David A Weitz, James P Butler, and Jeffrey J Fredberg. Physical forces during collective cell migration. *Nature physics*, 5(6):426–430, 2009.

[Vog23] Tim Vogel. Modelling of cells and their dynamics. 2023.

[VT66] RB Vaughan and JP Trinkaus. Movements of epithelial cell sheets in vitro. *Journal of cell science*, 1(4):407–413, 1966.

[WV21] Dennis Wenzel and Axel Voigt. Multiphase field models for collective cell migration. *Physical Review E*, 104(5):054410, 2021.