Scientific Annals of Computer Science vol. 20, 2010
"Alexandru Ioan Cuza" University of Iaşi, Romania

Shape Calculus.
A Spatial Mobile Calculus for 3D Shapes

Ezio BARTOCCI[1], Flavio CORRADINI[1], Maria Rita DI BERARDINI[1], Emanuela MERELLI[1], Luca TESEI[1]

Abstract

We present a bio-inspired calculus for describing 3D shapes moving in a space. A shape forms a 3D process when combined with a behaviour. Behaviours are specified with a timed CCS-like process algebra using a notion of channel to naturally model binding sites on the surface of shapes. The calculus embeds collision detection and response, binding of compatible 3D processes and split of composed 3D processes.

1 Introduction

The Shape Calculus has been inspired and motivated by systems biology. In a near future, systems biology will profoundly affect healthcare and medical science. The ultimate aim is to design and test "in-silico" drugs giving rise to individualised medicines that will take into account physiology and genetic profiles [18]. This implies the existence of detailed digital models of each human organ and, possibly, of the whole human body considering the human biological systems together. The advantages of performing in-silico experiments by simulating a model, instead of arranging expensive in-vivo or in-vitro experiments, are evident. But of course the models should be as faithful as possible to the *real system*.

[1]School of Science and Technology, Computer Science Division, University of Camerino Via Madonna delle Carceri 9, 62032, Camerino (MC), Italy.
Email: `name.surname@unicam.it`

A real system is characterized by many biological phenomena that are inherently multiscale, i.e. they are characterised by interactions involving different scales at the same time. Models for describing and simulating biological systems have comparable resolution regimes and work on different spatial and temporal scales: in the microscopic approach, molecular dynamics and Monte Carlo methods describe systems at the level of atoms or proteins while, in the macroscopic regime, continuum-based simulations model complete biological assemblies (but do not contain any explicit molecular information). Actually, a characteristic of biological complexity is the intimate connection that exists between different length and time scales - from the fast nanometre-length scale of molecules to the slow highly structured meter scale of the whole human body. For instance, subtle changes in molecular structure as a consequence of a single gene mutation can lead to catastrophic failure at the organ level, such as heart failure from re-entrant arrhythmias that lead to ventricular fibrillation. But information flows equally in the reverse direction: mechanoreceptors at the cell level sense the mechanical load on the musculoskeletal system and influence gene expression via signal transduction pathways [21].

The molecular level is surely the scale at which biological systems have been studied more intensively in the perspective of systems biology, so far. Within this field, Takahashi et al. underline, in [29], the importance of considering space when modelling cellular phenomena and in particular biochemical signal cascades. They also highlight that macromolecular crowding in a limited space can also deeply affect biochemical reactions in the cell. Since physical concepts like space occupancy, intra-cellular movement, contacts (collisions) and shape transformation determine biomolecular interactions and therefore cell life, there is the need to provide physical characteristics (shape, mass, size, position) to entities. They can be collocated in the continuum space, autonomously move and interact with their spatial neighbour/colliding entities, react accordingly to their specified behaviour to reproduce the *emerging behaviour* observable in in-vivo and in-vitro experiments as well as at a higher scale of the same model.

Using a particle-based approach (i.e. there are specific actors that represent individuals) and adding geometric information (e.g. space, shape) to a model not only makes it more faithful and close to real biological systems (at any scale), but also gives the possibility to represent different levels of the same biological system in a uniform way. This characteristic results to be very useful in the construction of multiscale models. The main idea is that at every level of representation, being it organ, tissue, cellular or molecular, it is always possible to model the system using the same concepts: individual, moving entities that interact in the 3D space by binding to form complex entities or just by transforming into different entities by a concept of "reaction". Some recent works [10, 9, 11] give evidence of the advantages and of the feasibility of this approach. The idea of a geometric particle-based environment for simulation started, in our group, some years ago [12, 13] in the context of the simulation of biochemical reactions without using the classical approach of Ordinary Differential Equations. The idea has then evolved towards the realization of a simulator prototype, called BioShape, that embodies spatial 3D information and shape-based interactive entities [30, 10].

During the development of BioShape, several questions arose about how 3D shapes should be considered, how motion should be associated to them and, most importantly, what kind of interactions should be considered among entities in order to reproduce typical scenarios of biological (but not only limited to them) systems at any scale. We found several solutions about the possible representation of 3D shapes and their movement in a space, mainly in the computer game world. In this and in related fields, the problem of reproducing/simulating virtual physical environments has been intensively studied and a lot of libraries and tools exist to manage space and to deal with the unavoidable problem of collision detection and collision response. We mainly refer to [17, 23] and to references therein for a first glance in this rich and articulated world. However, the possibility to import good techniques for managing the virtual environment solved only half of the problem. The specification of the behaviour of the autonomous entities, and the modalities on which this behaviours should be based, represented another difficult question to address.

Even if BioShape embodies the agent-based technology [16] that realizes the abstraction of autonomous agents, we felt the need of defining a language abstract enough to represent the main behaviours of the entities and having a formal semantics suitable to describe the evolution of the system. This consideration has been the main motivation that led to the definition of Shape Calculus. The introduction of a formal calculus gives a very precise characterization of the environment in which our simulations should run, with all the advantages that a formal semantics brings to the development of the simulator itself. Formal methods have been studied in the past to describe and analyse (complex) software systems. Thus, several models and languages exist for specifying systems - based on automata, process algebras and Petri nets - and several verification techniques - model checking and equivalence checking being the most famous ones - have been introduced for verifying their qualitative and quantitative properties. Some of the models include quantitative information such as time, probabilities and costs, and relative quantitative verification techniques exist. The definition of the Shape Calculus and its semantics is the first step towards the natural completion of the framework with formal verification techniques. The objective is to find the right abstractions and boundaries that permit the application of existing quantitative model checking or quantitative equivalence checking techniques to the evolution of a given network of Shape Calculus processes. Both the simulation approach of BioShape and the verification approach of the Shape Calculus can lead to interesting results when applied to a specific domain. Our long term objective is to coupled them to work in synergy towards the gaining of quantitative information about a model.

In the Shape Calculus, 3D processes are entities composed of a 3D shape and a dynamic behaviour. Processes are situated in a space, move accordingly to personalized motion laws (as, for instance, acceleration in a gravitational field or Brownian motion or attraction towards a biochemical gradient), collide and possibly bind with others processes becoming compound 3D processes. The binding depends on channels $\langle a, X\rangle$, derived from classical CCS [24] channels, where a is the channel name, intended as a type for binding certain species, and X is a certain region on the surface of the 3D process in which the channel is "active". The binding corresponds to communication on these channels. It occurs if and only if the surface of contact between the two 3D processes belongs to active channels whose names are co-names à la CCS. Compound processes can split weakly, by non-deterministically releasing a previously established bond, or "react", by splitting urgently in as many pieces as the products of the reaction are. If communication (i.e. binding) does not occur, the collision of two 3D processes is considered elastic, i.e. the shapes bounce and proceed independently.

This paper introduces the language of the Shape Calculus with the main aim of gently and incrementally present all its features and their relative semantics. We want to discuss the motivations that led us to the choices we made about the type and the nature of the operators of the language and it is our purpose to give enough glimpses in order to appreciate the great variety of scenarios that can be modelled with this language. A more

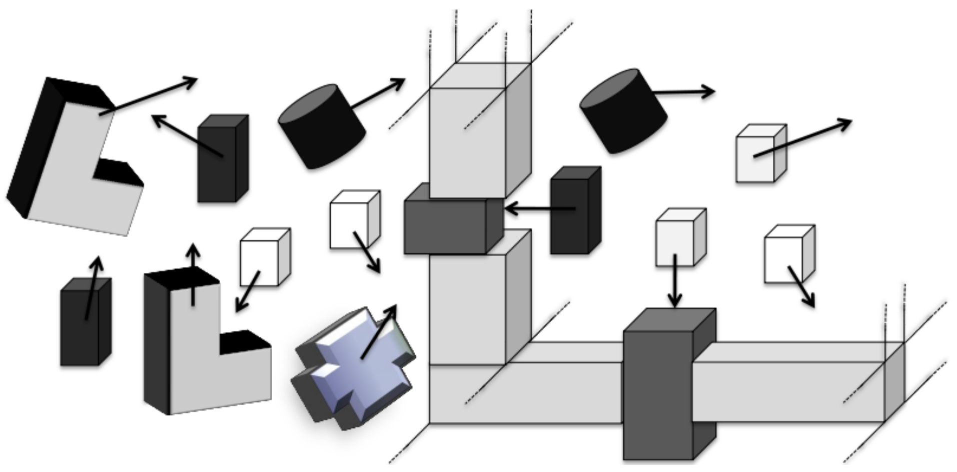

Figure 1: An example of a network of 3D processes

formal work on the timed operational semantics of the Shape Calculus and on a first global result of well-formedness of its possible dynamics will be available in a paper to be published, as the second part of this work. A short preliminary version of this paper appeared in [5, 6].

The paper is organised as follows. Section 2 gives an overview of the calculus combining graphical intuition with sample pieces of processes in order to understand the operators of the calculus and the reasons of their introduction. Section 3 introduces 3D shapes, shape composition, movement, collision detection and collision response. Section 4 defines behaviours and 3D processes while Section 5 puts all the pieces together and specifies networks of 3D processes. Finally, Section 6 presents related works and Section 7 concludes with ongoing and future work directions.

2 Overview of the Calculus

In this section we give some intuitions about the objects of the calculus and their possible behaviours. The general idea of the Shape Calculus is to consider a three-dimensional space in which several shapes reside, move and interact. Figure 1 shows a possible scenario at a certain time instant: on the left side there are simple 3D shapes (cubes, cylinders, etc.) or more complex ones, obtainable by "glueing" two given shapes on a common surface, moving freely in space. The arrows represent their instant velocity vectors. On the right side there is a composition of shapes enclosing a certain portion of the space. Their velocity vector is zero and it is intended to remain zero over time. These shapes can represent walls to the shapes inside and outside the enclosed region. Some of the pieces of the "walls" can represent doors that could open if some specific object hits them. We will call *network of 3D processes* a scenario like the one in the figure.

We want to stress that a network of 3D processes (3D network) can represent different biological systems at different scales. For instance, the space could be a portion of the cytoplasm of a cell in which some molecules, of different magnitudes, swim and interact by biochemical reactions. In this case walls can represent membranes encompassing compartments of the cell. However, the shapes can easily represent cells composing a tissue. In this case usually they do not move, but can interact by other shapes that are sent around as "shaped messengers". This cellular/tissue scenario has been adapted in [9] to shown how the phenomenon of bone remodelling can be easily modelled using shapes and their interactions. The tissue scale is represented by a 3D lattice of cubes, each of which again can be a 3D process, while in the cellular scale the specialized cells for bone absorption and reconstruction are modelled with proper (moving) 3D processes. The Shape Calculus can also be used to represent populations of animals, like fish or birds, and their dynamics and interactions in a given environment, as well as a completely different domain, at a very larger scale, such as astronomy: planets, comets, stars could be represented by shaped 3D processes that move in the cosmos guided by the law of gravitation. Of course, every model we can imagine has to cope with computational limits of simulation/verification. Experience teaches us that, in general, a particle-based 3D geometric approach, like the one we are proposing, finds a good compromise between computational feasibility and faithfulness of the model in cellular/tissue scenarios or in particular molecular scenarios. Consider, for instance, the case in which specific molecules are present and active only in some regions of the space (not well-stirred systems) or when there is a need of in-silico experiments in which additional molecules are to be added dynamically. It is well-known that, in these cases, ODE-based approaches fail because they implicitly assume well-stirredness, they lack of compositionality and they are static as the model has to be completely specified at the very beginning.

In this paper, we will mostly use biochemical reactions for giving in-

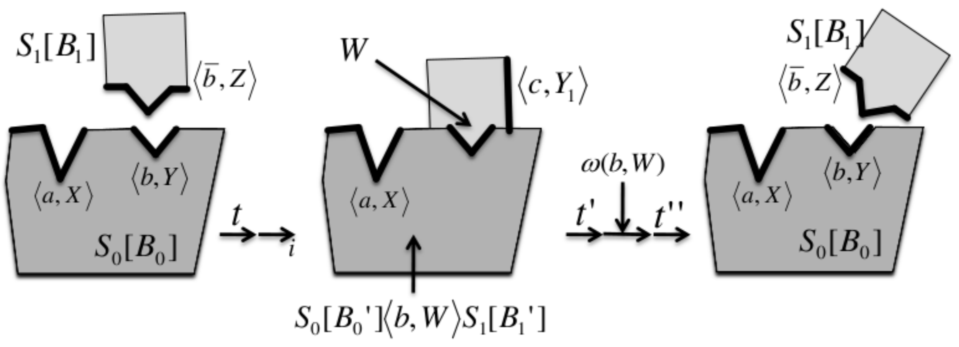

Figure 2: An example of binding and subsequent weak-split of two 3D processes

tuitions and examples. This will help us in introducing all the aspects of the calculus. Indeed, every species of molecule has a specific shape and we know from biology that the functions of a molecule are tightly related to its shape. For instance, in enzymatic reactions, the functional sites that are active in the enzyme structure, at a given time, determine which substrate (one or two metabolites) can bind the enzyme and proceed to the catalyzed reaction.

While time flows, shapes move according to their velocities that can change over time both due to a specific motion law - for instance as in a gravitational, electromagnetic, chemical attractive field, or Brownian motion - and due to collisions that can occur among shapes. Collisions can result in a bounce, i.e. they are considered *elastic* collisions. As it often happens in biology, colliding objects can bind and become a new, compound, object moving in a different way and possibly having a different behaviour. In this case we speak of "*inelastic*" collisions because they are treated with the physical law for that kind of collisions.

Figure 2 shows a (2D for simplicity) representation of a possible dynamics of an enzymatic reaction. Syntactically, we represent the larger shape, in this case playing the role of an enzyme, with the 3D process $S_0[B_0]$ with shape S_0 and behaviour B_0. The 3D process $S_1[B_1]$ represents a metabolite that is spatially close to the given enzyme. Note that some surfaces of the shapes are highlighted: they are the *channels* that the current 3D processes exhibit to the environment. Channels are specified in the behaviours of processes and consist of two components: a channel name and an active surface.

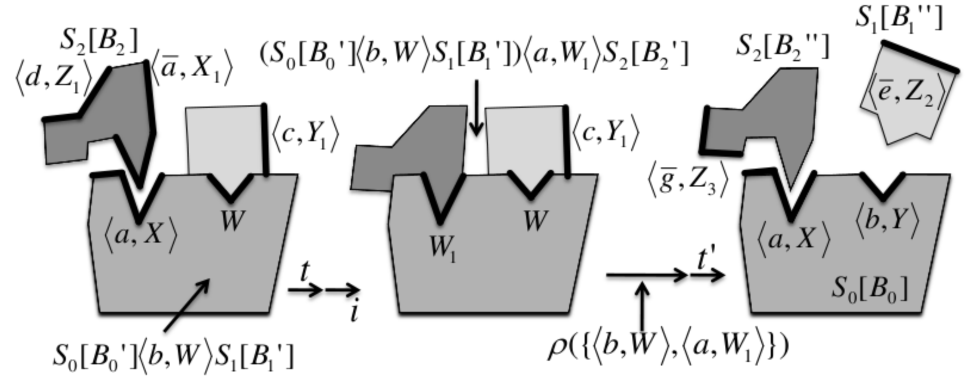
$S_2[B_2]$
$\langle d, Z_1\rangle$
$\langle \bar{a}, X_1\rangle$
$\langle c, Y_1\rangle$
$\langle a, X\rangle$
W
$\xrightarrow{t}\rightarrow_i$
$S_0[B_0']\langle b, W\rangle S_1[B_1']$
$(S_0[B_0']\langle b, W\rangle S_1[B_1'])\langle a, W_1\rangle S_2[B_2']$
$\langle c, Y_1\rangle$
W_1
W
$\xrightarrow{t'}$
$\rho(\{\langle b, W\rangle, \langle a, W_1\rangle\})$
$S_2[B_2'']$
$S_1[B_1'']$
$\langle \bar{e}, Z_2\rangle$
$\langle \bar{g}, Z_3\rangle$
$\langle a, X\rangle$
$\langle b, Y\rangle$
$S_0[B_0]$

Figure 3: An example of complex formation and subsequent strong-split

For instance, in Figure 2 $\langle a, X\rangle$ is an open channel, active on the site X, whose type is a. Note that $\langle a, X\rangle$ is a valid channel if X is a portion of the surface of S_0. In this case, the enzyme has two open channels and the process is specified as follows: $S_0[\langle b, Y\rangle.B_0' + \langle a, X\rangle.B_0'']$. The operator $+$ represents a non-deterministic choice between two alternative communication channels. This non-determinism is resolved during the evolution of the system depending on which 3D processes will collide with the enzyme and where.

Following the evolution proposed in the figure, after some time t elapsed (represented by the transition $\xrightarrow{t}$) and after a detection and resolution of an inelastic collision (transition $\rightarrow_i$), we get a compound process represented by $S_0[B_0']\langle b, W\rangle S_1[B_1']$, where $W = Y \cap Z$, i.e. the common surface of contact. Note that communication *is* the binding, and it can happen only if there is a collision between two processes that expose two compatible channels (name and co-name à la CCS[24]) on their common surface of contact. In this case, the common surface of contact on which the bond is established is called W. The name b is a memory for the type of channels that bound. If the channels were not compatible, the collision would have been treated as elastic and the two 3D processes would have simply bounced.

If we let, for instance, $B_1' = \langle c, Y_1\rangle.B_{other}$, the component whose shape is S_1 opens a new channel $\langle c, Y_1\rangle$. Since the behaviour of a composed 3D process is the interleaving of the behaviours of the components, the whole 3D process $S_0[B_0']\langle b, W\rangle S_1[B_1']$ does the same.

The third stage of Figure 2 represents the case in which a split occurs. Note that the behaviour of the processes returns to the initial situation. This evolution models naturally the behaviour of an enzyme binding with a substrate: it can happen for some reason that the bond is loose and the two molecules are free again. We call this event a *weak-split.* It is not an urgent event, thus it can be delayed of an unspecified time. This is another source of non-determinism in the calculus.

Figure 3 shows another possible evolution of $S_0[B'_0]\langle b, W\rangle S_1[B'_1]$ of Figure 2. In this case another substrate, process $S_2[B_2]$, binds - on its channel $\langle \overline{a}, X_1\rangle$ on the common surface W_1 - with the compound process. In the scenario of biochemical reactions, this means that a final complex has been formed, thus the reaction must proceed and the products must be released. For modelling this behaviour, the calculus provides a special event that we call *strong-split.* Differently from a weak-split, this event must occur as soon as it is enabled, i.e. when *all* the involved components can release *all* the involved bonds. In this example the involved components are $S_0[B'_0]$, $S_1[B'_1]$ and $S_2[B'_2]$ and the set of bonds is $L = \{\langle b, W\rangle, \langle a, W_1\rangle\}$. Note, finally, that the enzyme returns to its original state, while the metabolites that are released exhibit a different behaviour according to what they have become.

Our shapes are intended to move in space along time. One of the choices to be made for the calculus is how the velocity of each shape changes over time. We believe that a continuous updating of the velocity, that would be a candidate for an "as precise as possible" approach of modelling, is not a convenient choice. The main reason is the well-known compromise between the benefits of approximation and the complexity of precision. Our choice, also common in the computer graphics field [17], is to approximate a continuous trajectory of a shape with a polygonal chain, i.e. a piecewise linear curve in which each segment is the result of a movement with a constant velocity. The vertices of the chain corresponds to the updates of the velocity of the shape. To this purpose we define a global parameter $\Delta \in \mathbb{R}^+$, called *movement time step*, that represents the period of time after which the velocity of all shapes is updated. The quantification of Δ depends on the desired degree of approximation and also on other parameters connected to collision detection (see Section 3.2). The time domain $\mathbb{T} = \mathbb{R}_0^+$ is then divided into an infinite sequence of time steps t_i such that $t_0 = 0$ and $t_i \leq t_{i-1} + \Delta$ for all $i > 0$. Figure 4 shows a possible timeline with all relevant events. From t_0 to t_1 a full movement time step passes. This means that all the shapes moved with their assigned velocities for a time Δ and, during this

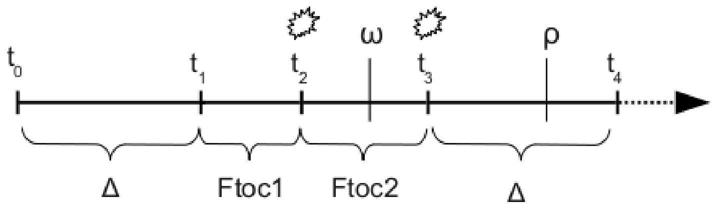

Figure 4: An example of timeline and the relative position of events

period, neither collisions nor splits (weak or strong) occurred. At the end of the period, the velocities of all the shapes are updated using the appropriate motion law (see Section 3.1). At this point the system would evolve of another full movement time step, but, from t_1 to t_2, this is not possible because a collision event is detected at a time $Ftoc1 < \Delta$. Thus, the system stops at t_2, which is $t_1 + Ftoc1$, resolves the collision event and updates the velocities of all the processes again. The same happens between t_2 and t_3, but in this case also a weak-split, represented by ω, occurs at a time instant in between. This event does not break the timeline with another t_i because there is no need to change any velocity of any shape. Simply, the 3D processes generated by the split physically still touch, but having the exact same velocity, do not collide. Only at time t_3, when their velocities will be updated, a possible collision can be detected. From t_3 to t_4 a full Δ can pass, meaning that no collisions are detected at time t_3 that occur before a time Δ. In this case, a strong-split, represented by ρ, happens at a time instant in between. Analogously to the weak-split, it does not break the timeline, but it is worth saying that a strong-split prevents time to pass further, i.e. it must be performed as soon as it is enabled, differently from a weak-split, that, even if enabled, can be delayed.

3 3D Shapes

Let us introduce three dimensional shapes as terms of a suitable language, allowing simpler shapes to bind and form more complex shapes. From now on, we consider assigned a *global* coordinate system in a three dimensional space represented by $\mathbb{R}^3$. Let $\mathbb{P}, \mathbb{V} = \mathbb{R}^3$ be the sets of positions and velocities, respectively, in this coordinate system.

For convenience we use, throughout the paper, relative coordinate systems that will always be w.r.t. a certain shape S, that is to say the origin of the relative system is a reference point $\mathbf{p}$ of S. We refer to this relative system as the *local* coordinate system of shape S. Given a point $\mathbf{p} \in \mathbb{P}$, expressed in the global coordinates, and a set of points $U \subseteq \mathbb{P}$, expressed in a local coordinate system whose origin is $\mathbf{p}$, we define $\mathsf{global}(U, \mathbf{p}) = U + \mathbf{p} = \{(\mathbf{u} + \mathbf{p}) \in \mathbb{P} \mid \mathbf{u} \in U\}$, i.e. the set of points U expressed in the global coordinates. Using the local system we can express parts of S - such as a certain face, a certain vertex, etc. - independently from the actual global position of the shape.

Definition 1 (Basic Shapes) *A basic shape σ is a tuple $\langle V, m, \mathbf{p}, \mathbf{v}\rangle$ where $V \subseteq \mathbb{P}$ is either a* sphere*, a* cone*, a* cylinder *or a* convex polyhedron[2]*, $m \in \mathbb{R}^+$ is the mass of the shape, $\mathbf{p} \in \mathbb{P}$ is the centre of mass[3] of the shape and $\mathbf{v} \in \mathbb{V}$ is the vector representing the current velocity of the shape.*

We define the following quantities on a basic shape σ: the points $\mathcal{P}(\sigma) = V$*, the* velocity $\mathbf{v}(\sigma) = \{\mathbf{v}\}$*, the* mass $\mathbf{m}(\sigma) = m$ *and the* reference point $\mathcal{R}(\sigma) = \mathbf{p}$*. The* boundary $\mathcal{B}(\sigma)$ *of σ is the subset of points of $\mathcal{P}(\sigma)$ that are on the surface of σ[4].*

The set of all possible basic shapes, ranged over by $\sigma, \sigma', \ldots$, is denoted by Basic.

Note that we use only very simple basic shapes that can be represented by suitable and efficient data structures and are handled by the most popular algorithms for motion simulation, collision detection and collision response [17]. Moreover, note that we consider only convex shapes. Recall that a set of points $U \subseteq \mathbb{P}$ is *convex* if and only if for every $\mathbf{x}$, $\mathbf{y}$ in U the set $\{(\lambda\mathbf{x} + (1 - \lambda)\mathbf{y}) \in \mathbb{P} \mid 0 \leq \lambda \leq 1\}$ is contained in U.

Three dimensional shapes of any form can be approximated with arbitrary precision by composing basic shapes in the following sense: the composition of two shapes corresponds to the construction of a third shape by "glueing" the two components on a common surface. Consider the shape shown in Figure 5(a). It is composed of the basic shape σ_1 "glued" with the basic shape σ_2 on the common surface X, called *surface of contact*. Note that no interpenetration between the composing shapes is allowed.

[2]From a syntactical representation point of view, we assume that V is finitely represented by a suitable data structure, such as a formula or a set of vertices.

[3]We actually need only a reference point. Thus, any other point in V can be chosen.

[4]Note that we consider only closed shapes, i.e. they contain their boundary.

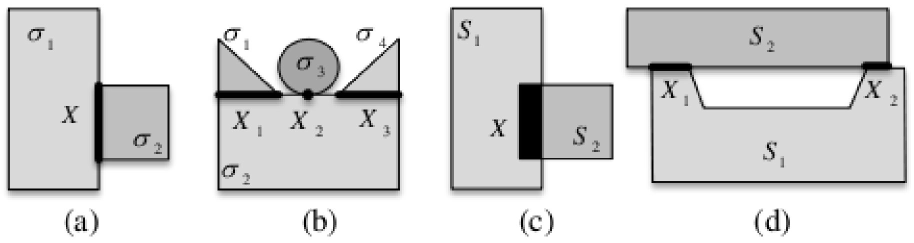

Figure 5: Some examples of 2D composed shapes

This concept can be generalised to the composition of two generic shapes either basic or compound under the same hypotheses, i.e. they bind on a common surface, but they do not interpenetrate.

Definition 2 (3D shapes) *The set* $\mathbb{S}$ *of* 3D shapes*, ranged over by* $S, S', \ldots$ *is generated by the grammar* $S ::= \sigma \bigm| S\,\langle X\rangle\,S$ *where* $\sigma \in \mathsf{Basic}$ *and* $X \subseteq \mathbb{P}$.

Starting from the same concepts defined for basic shapes, we inductively define the points, the velocity, the mass and the reference point of a compound shape $S = S_1\,\langle X\rangle\,S_2$ as $\mathcal{P}(S) = \mathcal{P}(S_1) \cup \mathcal{P}(S_2)$, $\mathbf{v}(S) = \mathbf{v}(S_1) \cup \mathbf{v}(S_2)$, $\mathbf{m}(S) = \mathbf{m}(S_1) + \mathbf{m}(S_2)$ and $\mathcal{R}(S) = (\mathbf{m}(S_1) \cdot \mathcal{R}(S_1) + \mathbf{m}(S_2) \cdot \mathcal{R}(S_2))/(\mathbf{m}(S_1) + \mathbf{m}(S_2))$[5]. The boundary of a compound shape S is defined as the surface of the resulting shape and is denoted by $\mathcal{B}(S)$. More formally, $\mathcal{B}(S_1\,\langle X\rangle\,S_2) = (\mathcal{B}(S_1) \cup \mathcal{B}(S_2)) \setminus \{\mathbf{x} \in \mathbb{P} \mid \mathbf{x} \text{ is interior of } \mathcal{P}(S_1\,\langle X\rangle\,S_2)\}$, where a point $\mathbf{x} \in U \subseteq \mathbb{P}$ is called *interior* of U if there exists an open ball with centre $\mathbf{x}$ which is completely contained in U.

In this paper we only consider *well-formed* 3D shapes, i.e. basic shapes or compound shapes in which X is a surface of contact, the components do not interpenetrate and they all must have the same velocity. The concept of touching without interpenetrating will be useful in the following when we define collision detection and compound 3D processes. By definition, X is always on the boundary of both S_1 and S_2. Thus, the set X can be a single point, a segment or a surface, depending on where the two shapes are touching without interpenetrating. Most of the time X is a (subset of a) *feature* of the basic shapes composing the 3D shape, i.e., a face, an edge or a vertex. Moreover, the fact that all the basic shapes forming a compound

[5]For simplicity, as above, we use the centre of mass as the reference point. Any other point can be chosen.

shape have the same velocity means that the compound shape moves as a unique body.

Figure 5 shows four examples of compound shapes. Note that while basic shapes are all convex, compound shapes can be non-convex, as those shown in figure. Shape in Figure 5(b) is composed of four basic shapes. A well-formed term representing this shape is $((\sigma_1 \langle X_1 \rangle \sigma_2) \langle X_2 \rangle \sigma_3) \langle X_3 \rangle \sigma_4$. Note, for instance, that X_1 is exactly the intersection $\mathcal{B}(\sigma_1) \cap \mathcal{B}(\sigma_2)$ and that is equal to one feature (an edge) of σ_1. The surface of contact X_2 contains only one point of contact and is subset of $\mathcal{B}(\sigma_1 \langle X_1 \rangle \sigma_2) \cap \mathcal{B}(\sigma_3)$, i.e. its two immediate sub-components. Figure 5(c) is an example of a not well-formed shape because there is interpenetration between S_1 and S_2. In Figure 5(d) there is an example of a well-formed shape in which the intersection of (the boundaries of) the two components S_1 and S_2, called X in the figure, is *not* a connected set. Recall that a set U is *connected* if and only if the only pair of disjoint closed sets whose union is U is the pair $(\emptyset, U)$. Note that if the intersection (the boundaries of) two compound shapes is not connected it is always a finite union of connected sets. In this case we obtain a shape with a hole. We admit such shapes in the calculus since they can be formed by correct bindings of well-formed shapes.

Given a 3D shape S, it can be represented by arranging the basic shapes and the surfaces of contact in different ways. A structural congruence $\equiv_S$ relation among terms representing shapes can be defined in a natural way. For instance, the shape of Figure 5(b) can be represented by structural congruent terms, e.g.
$((\sigma_3 \langle X_2 \rangle \sigma_2) \langle X_1 \rangle \sigma_1) \langle X_3 \rangle \sigma_4$ or $\sigma_1 \langle X_1 \rangle (\sigma_4 \langle X_3 \rangle (\sigma_3 \langle X_2 \rangle \sigma_2))$ etc.

Example 1 (A Biological Example) *Let us introduce an example of use of the parts of the calculus introduced so far. The* glycolysis *pathway is part of the process by which individual cells produce and consume nutrient molecules. It consists of ten sequential reactions, all catalyzed by a specific enzyme. Let us focus on the first reaction that can be described as*

$$\text{glucose, ATP} \quad \rightleftharpoons \quad \text{glucose-6-phosphate, ADP, H}^+$$

where an ATP is consumed and a molecule of glucose (GLC) is phosphorylated to glucose 6-phosphate (G6P), releasing an ADP molecule and a positive hydrogen ion (Hydron). The enzyme catalysing this first reaction is Hexokinase *(HEX). GLC, G6P, ATP, ADP and H^+ are metabolites. Both enzymes and metabolites are autonomous cellular entities that continuously move within the cytoplasm. The transformation of a metabolite into the one*

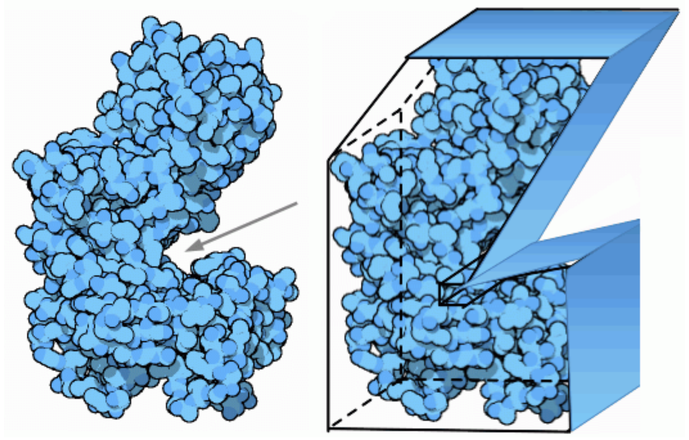
Figure 6: The real shape of the unbound Hexokinase and an approximation via a compound shape

that follows in the "pipeline" of the pathway (in this case, GLC into G6P) depends on the contact (collision in binding sites) of the right enzyme (in this example HEX) with the right metabolites, in this example GLC and ATP. The order of these bindings does not matter. After this binding, the reaction takes place and the products[6] *are released in the cytoplasm (i.e. a strong-split is performed). A special case occurs when the enzyme has bound one metabolite and an environmental event makes it release the metabolite and not proceed to the completion of the reaction (i.e. a weak-split is performed).*

We model the shape of HEX, which we denote S_h*, by a polyhedron approximating its real shape and mass (available at public databases (e.g. [2]). Figure 6 shows an example of such an approximation. Figure 7 shows a network of 3D processes in which there are two Hexokinase 3D processes and some GLC, G6P, ATP and ADP 3D processes. Note that we use a unique kind of shape for GLC ad G6P, denoted by* S_g*, and another unique kind of shape for ATP and ADP, denoted* S_a*. They can be distinguished only by their behaviours. Moreover, note that* S_g *and* S_a *are also approximations of the real shapes of the metabolite and the ratio of magnitude is respected. The other annotations present in the figure can be ignored for the moment. They will be used in Section 4 to further develop the glycolysis example.*

[6]In this example we neglect the hydron.

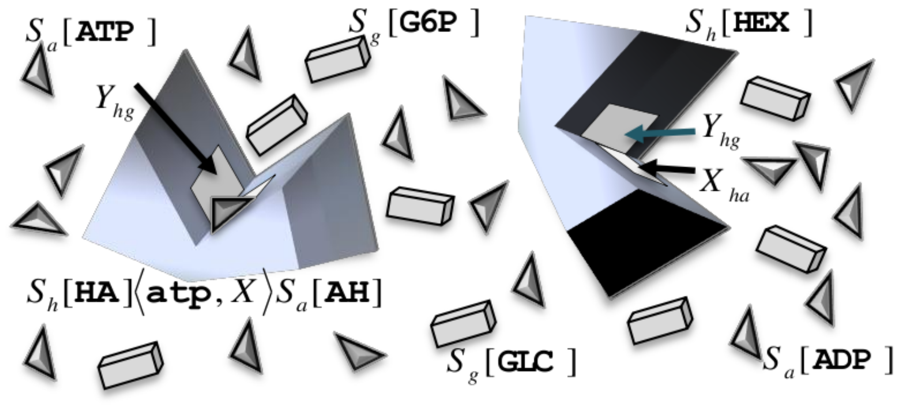

Figure 7: A network of 3D processes for describing the first reaction of the glycolysis pathway

3.1 Trajectories of Shapes

As mentioned above, given a well-formed term S, it represents shape at a certain time instant t. The velocity of S in that instant is $\mathbf{v}(\mathrm{S})$. The updating of the velocities occurs at every time instant t_i in which the timeline is broken up. In the calculus this updating is performed by exploiting a function

$$\mathsf{steer}\colon \mathbb{T} \to (\mathbb{S} \hookrightarrow \mathbb{V})$$

that defines how to change the velocity of all existing shapes (i.e. all shapes that are currently moving in the space) at each time t. We assume that, at any given time instant $t \in \mathbb{T}$, $\mathsf{steer}\, t\, S$ is undefined iff the shape S does not exist and, hence, its velocity has not to be changed. Note that this approach gives the maximal flexibility for defining motion. Static shapes can be represented by assigning always $\mathbf{0}$ to their velocity[7]. A gravity field can be simulated by updating velocities accordingly to the gravity acceleration (see Figure 8). A Brownian motion can be simulated by choosing a random direction in 3D and then defining the length of the vector w.r.t. the mass and/or the volume of the shape.

For the sake of simplicity we do not consider, in this paper, movements by rotations. However, this kind of movement can be easily added (it is

[7]If we want to represent for instance walls, we also need to assign an infinite value to the mass of these objects, otherwise they can be moved anyway due to collisions.

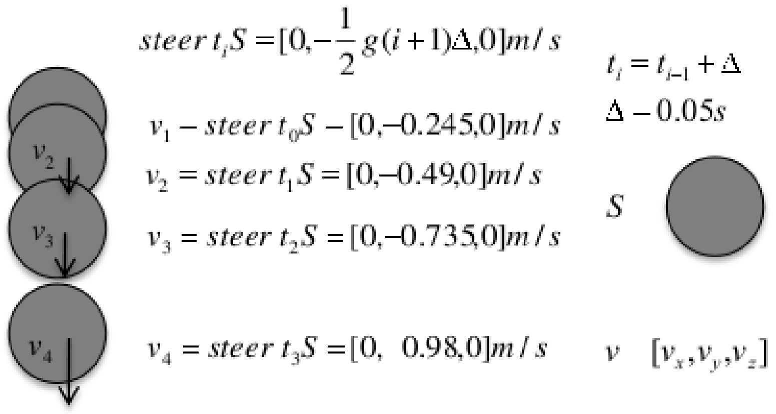
steer t_i S = [0, −1/2 g(i + 1)Δ, 0] m/s
t_i = t_{i−1} + Δ
Δ = 0.05s
v_1 = steer t_0 S = [0, −0.245, 0] m/s
v_2 = steer t_1 S = [0, −0.49, 0] m/s
v_3 = steer t_2 S = [0, −0.735, 0] m/s
v_4 = steer t_3 S = [0, 0.98, 0] m/s
v_2
v_3
v_4
S
v [v_x, v_y, v_z]

Figure 8: An example of updating of the velocity in presence of a gravitational field

present in BioShape) to our shapes by assigning an angular velocity and a moment of inertia to a shape and then by performing a compound motion of translation and rotation along the movement time step.

In Example 1, as we model the molecules at the mesoscale (10^{-8} - 10^{-7} m), Brownian motion is generally considered a good approximation for their motion. Thus, the three kind of shapes S_h, S_g and S_a all are subject to the Brownian updating of velocity.

3.2 Collision Detection

Our intent is to represent a lot of shapes moving simultaneously in space as described above. Inevitably, this scenario produces collisions between shapes when their trajectories meet. There is a rich literature on collision detection systems, as this problem is fundamental in popular applications like computer games. Good introductions to existing methods for efficient collision detection are available and we refer to Ericson [17] and references therein for a detailed treatment.

For our purposes, it is sufficient to define an interface between our calculus and a typical collision detection system. We can then choose one of them according to their different characteristics, e.g. their applicability in large-scale environments or their robustness. It must be said, however, that the choice of the collision detection system may influence the kind of

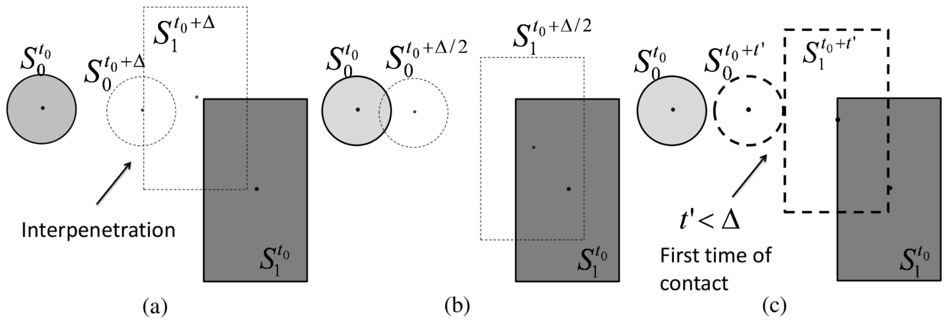

Figure 9: Some steps to determine the first time of contact between two shapes

basic (or compound) shapes we can use, as, for instance, some systems may require the use of only convex shapes to be more efficient[8].

Typically, a collision detection algorithm assumes to start in a situation in which shapes do not interpenetrate. Then it tries to move all the shapes of a little time step - that we have already introduced as movement time step Δ - and check if interpenetrations occurred[9]. If so, it tries to consider only half of the original time step and repeat the interpenetration check, i.e. it performs a binary search of the *first time of contact* t between two or more shapes, with some degree of approximation. Figure 9 shows these steps. In case (a) the passage of the whole Δ results in an interpenetration. Then, in (b) the passage of $\Delta/2$ is tried resulting into a non-contact. After some iterations the situation in (c) is reached.

In addition to the first time of contact, a collision detection algorithm usually outputs the shapes that are colliding, i.e. are touching without interpenetrating after t, and some information about the surfaces or points of contact.

Let I be a non-empty finite set of indexes and let $\{S_i\}_{i \in I}$ be a set of well-formed shapes such that for all $i, j \in I$, S_i and S_j do not interpenetrate (Def. 2). Roughly speaking, the *first time of contact* of the shapes

[8]The basic shapes that we consider in Definition 1 are typically accepted by most of the collision detection systems.

[9]Typically, the major efforts of optimisation are concentrated in this step since the number of checks is, in the worst case, $O(N^2)$ - where N is the number of shapes in the space - but the shapes that are likely to collide are only those that are very close to each other.

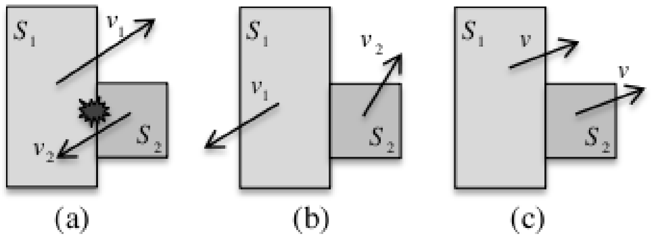

Figure 10: Some situations in which a collision is detected or not detected

S_i, denoted $\mathsf{Ftoc}(\{S_i\}_{i \in I})$, is the least number $t \in \mathbb{T}$ such that, moving shapes in $\{S_i\}_{i \in I}$ for t time units, a configuration is reached in which at least two shapes touch and do not interpenetrate, but an interpenetration occurs letting any other positive time ϵ elapse.

Note that such a definition allows shapes that are touching without interpenetrating and with velocities that do not make them to interpenetrate (e.g., the same velocity) to move without triggering a first time of contact. This possibility, as we mentioned in Section 2, is useful when we split previously compound shapes. Giving them the same velocity vector after the split, we are guaranteed that the first time of contact currently in force is not affected by the split. Figure 10(a) shows a situation in which a collision is detected, while in cases (b) and (c) no collision is detected because letting any time pass, the two shapes will not interpenetrate.

3.3 Collision Response

In this section, we briefly discuss the problem of *collisions response* [20], i.e. how collisions, once detected, can be resolved. In what follows we distinguish between *elastic* collisions (those in which there is no loss in kinetic energy) and *perfectly inelastic* ones (in which kinetic energy is fully dissipated)[10]. After an elastic collision, two shapes will proceed independently to each other but their velocities will change according to the laws for conservation of linear momentum and kinetic energy. On the other hand, as a consequence of an inelastic collision, two shapes will bind together and will move as a *unique* body whose velocity is determined by the law for conservation of linear momentum only.

[10]Other different kinds of collisions can be easily added to the calculus provided that the corresponding collision response laws are given.

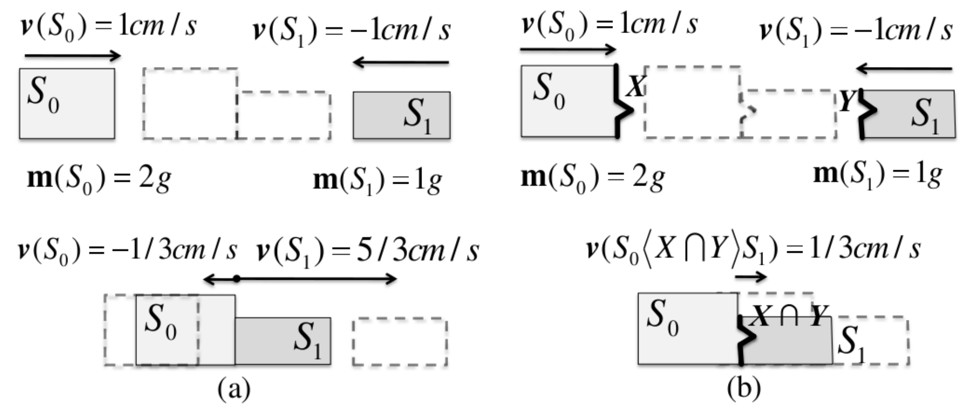

Figure 11: Examples of collision response of elastic and inelastic collision.

In Figure 11(a) it is shown an example of collision response to an elastic collision along only one dimension. In Figure 11(b) there is another example of an inelastic collision.

4 3D Processes

In this section we introduce the timed process algebra whose terms describe the *internal behaviour* of 3D shapes. This is a variation of Timed CCS [31], where basic actions provide information about binding capability and split of shape bonds.

We use the following notation. Let $\Lambda = \{a, b, \cdots\}$ be a countably infinite set of *channels names* (names, for short) and $\overline{\Lambda} = \{\overline{a} \,|\, a \in \Lambda\}$ its complementation. Let $\mathcal{A} = \Lambda \cup \overline{\Lambda}$; by convention we assume $\overline{\overline{a}} = a$ for each name a. Elements in $\mathcal{A}$ are ranged over by $\alpha, \beta, \cdots$.

Binding capabilities are represented by *channels*, i.e. pairs $\langle \alpha, X\rangle$ where $\alpha \in \mathcal{A}$ is a name and X is a *surface of contact*. Intuitively, a surface of contact is a portion of space (usually a subset of the boundary of a given 3D shape) where the channel itself is active and where binding with other processes are possible. Names introduce a notion of compatibility between channels that we will use in Section 5 to distinguish between elastic and inelastic collisions. We say that channels $\langle \alpha, X\rangle$ and $\langle \beta, Y\rangle$ are *compatible*, written $\langle \alpha, X\rangle \sim \langle \beta, Y\rangle$, if $\beta = \overline{\alpha}$ and $X \cap Y \neq \emptyset$. Otherwise, $\langle \alpha, X\rangle$ and $\langle \beta, Y\rangle$ are said to be *incompatible* (that we write as $\langle \alpha, X\rangle \not\sim \langle \beta, Y\rangle$). We also introduce two different kinds of actions that represent split of shape bounds. We distinguish between weak-split actions $\omega(\alpha, X)$ and strong-split actions $\rho(\alpha, X)$ where $\langle \alpha, X \rangle$ is a channel. With an abuse of notation, we say that two weak-split actions $\omega(\alpha, X)$ and $\omega(\beta, Y)$ (as well as two strong-split actions $\rho(\alpha, X)$ and $\rho(\beta, Y)$) are compatible if so are the channels $\langle \alpha, X \rangle$ and $\langle \beta, Y \rangle$.

As we will see later on in this paper, synchronisations between a pair of compatible weak-split actions result in a weak-split operation, while synchronisations between multiple pairs of compatible strong-split actions correspond to a strong-split operation. These operations behave differently w.r.t. to *time passing* since the latter operation cannot let time pass further, while the former one can be arbitrarily delayed.

Let $\mathcal{C}$ be the set of all channels, $\omega(\mathcal{C}) = \{\omega(\alpha, X) \mid \langle \alpha, X \rangle \in \mathcal{C}\}$ and $\rho(\mathcal{C}) = \{\rho(\alpha, X) \mid \langle \alpha, X \rangle \in \mathcal{C}\}$ be the sets of weak-split actions and strong-split actions, respectively.

Our processes perform basic and *atomic* actions that belong to the set $\mathsf{Act} = \mathcal{C} \cup \omega(\mathcal{C}) \cup \rho(\mathcal{C})$ whose elements are ranged over by $\mu, \mu', \cdots$. We finally assume a countably infinite collection $\mathcal{K}$ of *process name* or *process constants.*

Definition 3 (Shape behaviours) *The set $\mathbb{B}$ of* shape behaviours *is generated by the following grammar*

$$B ::= \mathsf{nil} \mid \langle \alpha, X \rangle.B \mid \omega(\alpha, X).B \mid \rho(L).B \mid \epsilon(t).B \mid B + B \mid K$$

where $\langle \alpha, X \rangle \in \mathcal{C}$, $L \subseteq \mathcal{C}$ (non-empty) whose elements are pairwise incompatible (i.e. for each pair $\langle \alpha, X \rangle, \langle \beta, Y \rangle \in L$ it is $\langle \alpha, X \rangle \not\sim \langle \beta, Y \rangle$), $t \in \mathbb{T}$ and K is a process name in $\mathcal{K}$.

A brief description of our operators now follows. nil is the Nil-behaviour, it cannot perform any action but can let time pass without limits. A trailing nil will often be omitted, so e.g. we write $\langle a, X \rangle.\omega(a, X)$ to abbreviate $\langle a, X \rangle.\omega(a, X).\mathsf{nil}$. $\langle \alpha, X \rangle.B$ and $\omega(\alpha, X).B$ are action-prefixing known from CCS; they evolve in B by performing, resp., the actions $\langle \alpha, X \rangle$ and $\omega(\alpha, X)$. A behaviour of the form $\langle \alpha, X \rangle.B$ exhibits a binding capability along the channel $\langle \alpha, X \rangle$, while $\omega(\alpha, X).B$ models the behaviour of a shape that, before evolving in B, wants to split a *single* bond established via the channel $\langle \alpha, X \rangle$. $\rho(L).B$ is the *strong-split operator*; it can evolve in B only after that *all* bonds established along channels in L have been split. The delay-prefixing operator $\epsilon(t).B$ (see [31]) introduces time delays in 3D processes;

$$\textsc{Nil}_t\frac{}{\mathsf{nil} \overset{t}{\rightsquigarrow} \mathsf{nil}} \qquad \textsc{Pref}_t\frac{\mu \in \mathcal{C} \cup \omega(\mathcal{C})}{\mu.B \overset{t}{\rightsquigarrow} \mu.B} \qquad \textsc{Str}_t\frac{}{\rho(L).B \overset{t}{\rightsquigarrow} \rho(L).B}$$

$$\textsc{Sum}_t\frac{B_1 \overset{t}{\rightsquigarrow} B_1' \quad B_2 \overset{t}{\rightsquigarrow} B_2'}{B_1 + B_2 \xrightarrow{t} B_1' + B_2'} \qquad \textsc{Del}_t\frac{t' \geq t}{\epsilon(t').B \overset{t}{\rightsquigarrow} \epsilon(t'-t).B}$$

$$\textsc{Def}_t\frac{B \overset{t}{\rightsquigarrow} B'}{K \overset{t}{\rightsquigarrow} B'} \quad \text{if } K \overset{\text{def}}{=} B$$

Table 1: Temporal behaviour of $\mathbb{B}$ terms

$t \in \mathbb{T}$ is the amount of time that has to elapse before the idling time is over. Finally, $B_1 + B_2$ models a non-deterministic choice and K is a process definition.

In the remainder of this paper, we use processes in $\mathbb{B}$ to define the internal behaviour of our 3D shapes. For this reason, we assume that sites in binding capabilities, as well as in weak- and strong-split actions, are expressed w.r.t. a *local* coordinate system whose origin is the reference point of the shape where they are embedded in.

Definition 4 (Timed operational semantics of shape behaviours) *The SOS-rules defining the* temporal transitions *relations* $\overset{t}{\rightsquigarrow} \subseteq (\mathbb{B} \times \mathbb{B})$ *for* $t \in \mathbb{T}$ *are provided in Table 1. These transitions describe how processes in* $\mathbb{B}$ *evolve by letting time t pass. As usual, we write* $B \overset{t}{\rightsquigarrow} B'$ *if* $(B, B') \in \xrightarrow{t}$ *and* $B \overset{t}{\rightsquigarrow}$ *if there is* $B' \in \mathbb{B}$ *such that* $(B, B') \in \overset{t}{\rightsquigarrow}$*. Similar conventions will apply later on. Table 2 provides the SOS-rules defining the* action transitions *relations* $\xrightarrow{\mu} \subseteq (\mathbb{B} \times \mathbb{B})$ *for* $\mu \in \mathsf{Act}$*. These transitions describe which basic actions a shape's behaviour can perform.*

Most of the rules in Table 1 are those provided in [31]. Rules $\textsc{Pref}_t$ and $\textsc{Str}_t$ state that processes of the form $\langle \alpha, X \rangle.B$, $\omega(\alpha, X).B$ and $\rho(L).B$ can be arbitrarily delayed.

The only rules in Table 2 worth noting are those defining the functional behaviour of the strong-split operator. By Rules $\textsc{Str}_1$ and $\textsc{Str}_2$ $\langle \alpha, X \rangle \in L$ implies that $\rho(L).B$ can do a $\rho(\alpha, X)$-action and evolve either in B (if

$$\textsc{Pref}_a\frac{\mu \in \mathcal{C} \cup \omega(\mathcal{C})}{\mu.B \xrightarrow{\mu} B} \qquad \textsc{Del}_a\frac{B \xrightarrow{\mu} B'}{\epsilon(0).B \xrightarrow{\mu} B'} \qquad \textsc{Sum}_a\frac{B_1 \xrightarrow{\mu} B'}{B_1 + B_2 \xrightarrow{\mu} B'}$$

$$\textsc{Def}_a\frac{B \xrightarrow{\mu} B'}{K \xrightarrow{\mu} B'} \;\text{ if } K \overset{\text{def}}{=} B \qquad \textsc{Spt}_1\frac{L = \{\langle \alpha, X\rangle\}}{\rho(L).B \xrightarrow{\rho(\alpha,X)} B}$$

$$\textsc{Str}_2\frac{L = \{\langle \alpha, X\rangle\} \cup L' \quad L' \neq \emptyset}{\rho(L).B \xrightarrow{\rho(\alpha,X)} \rho(L').B} \qquad \textsc{Str}_3\frac{B \xrightarrow{\rho(\alpha,X)} B'}{\rho(L).B \xrightarrow{\rho(\alpha,X)} \rho(L).B'}$$

Table 2: Functional behaviour of $\mathbb{B}$ terms

$L = \{\langle \alpha, X\rangle\}$) or in $\rho(L\backslash\{\langle \alpha, X\rangle\}).B$ (otherwise). Rule $\textsc{Str}_3$ is needed to handle arbitrarily nested terms, e.g. $\rho(\{\langle a, X\rangle\}).\rho(\{\langle \bar{b}, Y\rangle\}).B$. Other rules are as expected.

Now we are ready to define our 3D processes that are simply or compound shapes with a given behaviour expressed as a $\mathbb{B}$-term.

Definition 5 (3D processes) *The set* $\mathsf{3DP}$ *of* 3D processes *is generated by the following grammar:* $P ::= S[B] \;\big|\; P\,\langle a, X\rangle\,P$ *where* $S \in \mathbb{S}$, $B \in \mathbb{B}$, $a \in \Lambda$ *and* X *is a non-empty subset of* $\mathbb{P}$.

The shape of each $P \in \mathsf{3DP}$ *is defined by induction on* P *as follows:*

Basic: $\mathsf{shape}(S[B]) = S$
Comp: $\mathsf{shape}(P\,\langle a, X\rangle\,Q) = \mathsf{shape}(P)\,\langle X\rangle\,\mathsf{shape}(Q)$

We define $\mathbf{v}(P) = \mathbf{v}(\mathsf{shape}(P))$ *and* $\mathcal{B}(P) = \mathcal{B}(\mathsf{shape}(P))$. *Finally,* $P[\![\mathbf{v}]\!]$ *is the 3D process we obtain by updating* P*'s velocity as follows:*

Basic: $(S[B])[\![\mathbf{v}]\!] = (S[\![\mathbf{v}]\!])[B]$
Comp: $(P\,\langle a, X\rangle\,Q)[\![\mathbf{v}]\!] = (P[\![\mathbf{v}]\!])\,\langle a, X\rangle\,(Q[\![\mathbf{v}]\!])$

We also write $\mathsf{steer}\,t\,P$ *to denote* $P[\![\mathsf{steer}\,t\,\mathsf{shape}(P)]\!]$.

Let us model the molecules involved in the reaction of Section 1 as 3D processes.

Example 2 (3D Processes for HEX, GLC and ATP) *Hexokinase can be modelled as* $HEX \overset{\text{def}}{=} S_h[\mathsf{HEX}]$ *where:*

$\mathsf{HEX} = \langle \mathsf{atp}, X_{ha}\rangle.\mathsf{HA} + \langle \mathsf{glc}, X_{hg}\rangle.\mathsf{HG}$,
$\mathsf{HA} = \omega(\mathsf{atp}, X_{ha}).\mathsf{HEX} + \epsilon(t_h).\langle \mathsf{glc}, X_{hg}\rangle.\rho(\{\langle \mathsf{atp}, X_{ha}\rangle, \langle \mathsf{glc}, Y_{hg}\rangle\}).\mathsf{HEX}$,
$\mathsf{HG} = \omega(\mathsf{glc}, X_{hg}).\mathsf{HEX} + \epsilon(t_h).\langle \mathsf{atp}, X_{ha}\rangle.\rho(\{\langle \mathsf{atp}, X_{ha}\rangle, \langle \mathsf{glc}, Y_{hg}\rangle\}).\mathsf{HEX}$,
and X_{ha}, Y_{hg} are the surfaces of contact shown in Figure 7. $ATP = S_a[\mathsf{ATP}]$ models an ATP molecule where:

$$\mathsf{ATP} = \langle \overline{\mathsf{atp}}, X_{ah}\rangle.(\epsilon(t_a).\rho(\{\langle \overline{\mathsf{atp}}, X_{ah}\rangle\}).\mathsf{ADP} + \omega(\overline{\mathsf{atp}}, X_{ah}).\mathsf{ATP})$$

and the surface of contact X_{ah} is the whole boundary $\mathcal{B}(S_a)$. The process modelling a molecule of glucose is similar: $GLC = S_g[\mathsf{GLC}]$ where

$$\mathsf{GLC} = \langle \overline{\mathsf{glc}}, X_{gh}\rangle.(\epsilon(t_g).\rho(\{\langle \overline{\mathsf{glc}}, X_{gh}\rangle\}).\mathsf{G6P} + \omega(\overline{\mathsf{glc}}, X_{gh}).\mathsf{GLC}$$

We leave unspecified the behaviours G6P *and* ADP.

HEX *exhibits two binding capabilities along the channels $\langle \mathsf{atp}, X_{ha}\rangle$ and $\langle \mathsf{glc}, Y_{hg}\rangle$ with and ATP- and a GLC-molecule, respectively. By performing an $\langle \mathsf{atp}, X_{ha}\rangle$-action*[11], HEX *evolves in* HA *that can do either an action $\omega(\mathsf{atp}, X_{ha})$ - and hence come back to* HEX *- or wait t_h units of time, perform $\langle \mathsf{glc}, Y_{hg}\rangle$*[12] *and then evolve in $\rho(\{\langle \mathsf{atp}, X_{ha}\rangle, \langle \mathsf{glc}, Y_{hg}\rangle\}).\mathsf{HEX}$. At this stage, we can perform the strong-split actions $\rho(\mathsf{atp}, X_{ha})$, $\rho(\mathsf{glc}, Y_{hg})$ and return to* HEX. *Notice that, after an $\langle \mathsf{glc}, Y_{hg}\rangle$-action,* HEX *becomes* HG *that behaves symmetrically.*

ATP *performs a $\langle \overline{\mathsf{atp}}, X_{ah}\rangle$-action, wait t_a units of time, and then can release the bond established on the channel $\langle \overline{\mathsf{atp}}, X_{ah}\rangle$ – and thus return free as ATP – or participate in the reaction and become an ADP. As we will see in Section 5, the result is the split of the complex in the three original shapes whose behaviours are* HEX, ADP *and* G6P, *respectively. We omit the description of the behaviour of* GLC *since it is similar to that of* ATP.

We are now ready to define the timed operational semantics of 3D processes.

Definition 6 (Weak timed operational semantics of 3D processes) *Rules in Table 3 define the transition relations $\overset{t}{\rightsquigarrow} \subseteq (\mathsf{3DP} \times \mathsf{3DP})$ for $t \in \mathbb{T}$, and $\xrightarrow{\mu} \subseteq (\mathsf{3DP} \times \mathsf{3DP})$ for $\mu \in \mathsf{Act}$. We have omitted symmetric rules for* COMP$_{a1}$ *and* COMP$_{a2}$ *for the actions of Q. Two 3D processes P and Q are said to be* compatible, *written $P \sim Q$, if $P \xrightarrow{\langle\alpha,X\rangle}$ and $Q \xrightarrow{\langle\overline{\alpha},Y\rangle}$ for some compatible channels $\langle\alpha,X\rangle, \langle\overline{\alpha},Y\rangle$; otherwise, P and Q are* incompatible *that we denote with $P \not\sim Q$. Below, we often write $P \overset{\rho}{\not\to}$ and $P \overset{\omega}{\not\to}$ as a shorthand for $P \overset{\rho(\alpha,X)}{\not\longrightarrow}$ and $P \overset{\omega(\alpha,X)}{\not\longrightarrow}$, resp., for any $\langle\alpha,X\rangle$*

[11]Whenever HEX and ATP collide, this action corresponds to a 'real' binding.
[12]This means that HEX can also bind with a colliding GLC-molecule.

$$\textsc{Basic}_t \frac{Bw \xrightarrow{t} B'}{S[B] \overset{t}{\rightsquigarrow} (S+t)[B']}$$

$$\textsc{Com}_t \frac{P \overset{t}{\rightsquigarrow} P' \quad Q \overset{t}{\rightsquigarrow} Q' \quad X' = X + (t \cdot \mathbf{v}(P))}{P \langle a, X \rangle Q \overset{t}{\rightsquigarrow} P' \langle a, X' \rangle Q'}$$

$$\textsc{Basic}_c \frac{B \xrightarrow{\langle \alpha, X \rangle} B' \qquad Y = \mathsf{global}(X, \mathcal{R}(S))}{S[B] \xrightarrow{\langle \alpha, Y \rangle} S[B']}$$

$$\textsc{Basic}_w \frac{B \xrightarrow{\omega(\alpha, X)} B' \qquad Y = \mathsf{global}(X, \mathcal{R}(S))}{S[B] \xrightarrow{\omega(\alpha, Y)} S[B']}$$

$$\textsc{Basic}_s \frac{B \xrightarrow{\rho(\alpha, X)} B' \qquad Y = \mathsf{global}(X, \mathcal{R}(S))}{S[B] \xrightarrow{\rho(\alpha, Y)} S[B']}$$

$$\textsc{Comp}_{a1} \frac{\mu \in \omega(\mathcal{C}) \cup \rho(\mathcal{C}) \qquad P \xrightarrow{\mu} P'}{P \langle a, X \rangle Q \xrightarrow{\mu} P' \langle a, X \rangle Q}$$

$$\textsc{Comp}_{a2} \frac{P \xrightarrow{\langle \alpha, Y \rangle} P' \qquad Y \subseteq \mathcal{B}(P \langle a, X \rangle Q)}{P \langle a, X \rangle Q \xrightarrow{\langle \alpha, Y \rangle} P' \langle a, X \rangle Q}$$

Table 3: Functional and temporal behaviour of 3DP terms

Rules in Table 3 say that a 3D process inherits its functional and temporal behaviour from the $\mathbb{B}$-processes we use define its internal behaviour. Notice that now sites of binding capabilities and split actions are expressed w.r.t. a *global* coordinate system (by rules $\textsc{Basic}_c$, $\textsc{Basic}_w$ and $\textsc{Basic}_s$). It is also noteworthy that, due rule $\textsc{Comp}_{a2}$, some of the $\langle \alpha, Y \rangle$-actions performed by either P or Q can be prevented in $P \langle a, X \rangle Q$ since, due to binding, the surface of contact $Y \not\subseteq \mathcal{B}(P \langle a, X \rangle Q)$ and, hence the corresponding channel is no more active.

Note that the operational semantics is called weak because it lets time pass even if a strong-split event is enabled. Later on, we will restrict this weak behaviour.

The operational rules in Table 3 do not allow synchronisation between components of compound process that proceed independently to each other. To illustrate this with an example, let us consider $P\,\langle a, X\rangle\,Q$ where $P = S_p[\rho(\{a, X_p\}).B_p]$, $Q = S_q[\rho(\{\overline{a}, X_q\}).B_q]$, $X = X'_p \cap X'_q$ and, for each $i \in \{p, q\}$, X'_i is the site X_i w.r.t. a global coordinate system, i.e. $X'_i = \mathsf{global}(X_i, \mathcal{R}(S_i))$. As stand-alone processes, P and Q can perform two compatible strong-split actions, namely $\rho(a, X'_p)$ and $\rho(\overline{a}, X'_q)$ and evolve, resp., in $S_p[B_p]$ and $S_q[B_q]$. As a consequence, $P\,\langle a, X\rangle\,Q$ becomes either $S_p[B_p]\,\langle a, X\rangle\,Q$ or $P\,\langle a, X\rangle\,S_p[B_q]$. According to the intuition given so far, P and Q have to synchronise on the execution $\rho(a, X'_p)$ and $\rho(\overline{a}, X'_q)$ in order to split the bond $\langle a, X\rangle$. This strong-split operation must produce two independent 3D processes, i.e. the *network* of 3D processes (see Section 5) that contains both $S_p[B_p]$ and $S_q[B_q]$. Similarly, weak-split operation are due to synchronisations on compatible weak-split actions. We roughly describe how to deal with this kind of behaviours. We first allow synchronisation on compatible split actions by introducing proper transition relations $\overset{\rho(a,X)}{\Rightarrow}$ and $\overset{\omega(a,X)}{\Rightarrow}$. Intuitively, we want that $P\,\langle a, X\rangle\,Q \overset{\rho(a,X)}{\Rightarrow} S_p[B_p]\,\langle a, X\rangle\,S_q[B_q]$. Now, to 'physically' remove the bond $\langle a, X\rangle$, we can define a function split that, given a 3D process and a set C of shape bonds (e.g. $\{\langle a, X\rangle\}$), removes all bonds in C and return the network of processes we are interested in.

We also recall that strong-split events can require simultaneous splits of multiple bonds. In this case, all the components involved in the reaction must - *all together* - be ready to synchronise on a proper set of compatible strong-split actions. Consider e.g. a more complicated example $(P\,\langle a, X\rangle\,Q)\,\langle b, Y\rangle\,R$ where $P = S_p[\rho(\{\langle a, X_p\rangle, \langle\overline{b}, Y_p\rangle\}).B_p]$, $Q = S_q[\rho(\{\langle\overline{a}, X_q\rangle\}).B_q]$, $R = S_r[\rho(\{\langle b, Y_r\rangle\}).B_r]$, $X = X'_p \cap X'_q$ and $Y = Y'_p \cap Y'_r$. We say that $(P\,\langle a, X\rangle\,Q)\,\langle b, Y\rangle\,R$ is *able to complete a reaction* to denote that it can satisfy all 'pending strong-split requests', and that the corresponding strong-split event is *enabled.* Recall that enabled strong-split events prevent the passage of time. To force this kind of timed behaviour we say that $P \xrightarrow{t} Q$ iff $P \overset{t}{\rightsquigarrow} Q$ and no strong-split event is enabled in in P. Due to this restricted timed behaviour, $(P\,\langle a, X\rangle\,Q)\,\langle b, Y\rangle\,R$ cannot let time pass. As a consequence, P can only synchronize with Q and R and split the bonds $\langle a, X\rangle$ and $\langle b, Y\rangle$ (the order does not matter). These two actions together correspond to a strong-split event that, once performed as a unique action, make the whole system able to continue its evolution also from the time passing point of view.

5 Networks of 3D processes

Now we can define a network of 3D processes as a collection of 3D processes that freely moving in the same 3D space.

Definition 7 (Networks of 3D processes) *The set* $\mathbb{N}$ *of networks of 3D processes is generated by the grammar*

$$N ::= \mathsf{Nil} \mid P \mid N \parallel N$$

where $P \in \mathsf{3DP}$. *We say that a network* N *is well-formed iff each 3D process composing the network is well-formed and, for each pair of distinct processes* P *and* Q *in the network,* $\mathsf{shape}(P)$ *and* $\mathsf{shape}(Q)$ do not interpenetrate. *Moreover, we extend the definition of* steer *on networks in the natural way, i.e. such that each process of the network is updated simultaneously.*

In our running example we construct a network of processes containing a proper number of HEX, ATP and GLC processes in order to replicate the conditions in a portion of cytoplasm.

Now, we define the temporal and functional behaviour of networks of 3D processes. Here, we assume that a network of 3D processes performs basic actions that belong to set $\{\omega, \rho\}$, where we use ω and ρ to denote, respectively, weak- and a strong-split of process bonds as a unique action (at a network level we only see if shape bonds can be split or not). In the following, we also let elements of the set $\{\omega, \rho\} \cup \mathbb{T}$ to be ranged over by $\nu, \nu', \cdots$.

Definition 8 (Temporal and Functional Behaviour of $\mathbb{N}$-terms) *Rules in Table 4 (plus an additional rule symmetric of* $\mathrm{P{\scriptsize AR}}_a$ *for actions of* M*) defines the transition relations* $\xrightarrow{t} \subseteq \mathbb{N} \times \mathbb{N}$ *for* $t \in \mathbb{T}$ *and* $\xrightarrow{\mu} \subseteq \mathbb{N} \times \mathbb{N}$ *for* $\mu \in \{\omega, \rho\}$. *A* timed trace *from a net* N *is a finite sequence of steps of the form*

$$N = N_0 \xrightarrow{\nu_1} N_1 \xrightarrow{\nu_2} \cdots \xrightarrow{\nu_n} N_n = M$$

We finally write that $N \overset{t}{\Rightarrow} M$ *if there exists a timed trace* $N = N_0 \xrightarrow{\nu_1} N_1 \xrightarrow{\nu_2} \cdots \xrightarrow{\nu_n} N_n = M$ *such that* $t = \sum_{i=0}^{n} \{\nu_i \mid \nu_i \in \mathbb{R}^+\}$.

A proper semantics that simply applies the rules to resolve collisions (both elastic and inelastic ones) can now be defined. It can then be used to formally specify the evolution of a network of 3D processes according to the timeline scheme that we sketched in Section 2.

$$\textsc{Empty}_t \frac{}{\mathsf{Nil} \xrightarrow{t} \mathsf{Nil}} \quad \textsc{Par}_t \frac{N \xrightarrow{t} N' \quad M \xrightarrow{t} M'}{N \parallel M \xrightarrow{t} N' \parallel M'} \quad \textsc{Par}_a \frac{N \xrightarrow{\mu} N'}{N \parallel M \xrightarrow{\mu} N' \parallel M}$$

Table 4: Temporal and functional behaviour of networks of 3D processes

6 Related works

In this section we give an overview of the modelling and simulation languages and tools that we consider connected with our approach in the area of formal calculi and of simulation tools for biological systems.

Many process algebras have been proposed in systems biology for modelling biological systems [26, 27, 14, 7, 15], accomplishing different kinds of abstractions. The common assumption in these calculi is that the systems are always well-stirred, which means that the positions of entities become randomly uniform over a contained volume. This distribution is often generated by several simulation methods [19] based on the theory of stochastic chemical kinetics. When systems are not well-stirred, the ideal way to simulate the time evolution of a chemical system would be to use molecular dynamics, in which the exact positions and velocities of all the molecules in the systems are tracked. In these cases the concepts of space and time play a fundamental role, and only recently they have been taken into account in process calculi for systems biology. BioAmbients [27] considers space as a set of communicating compartments, while in Spatial CLS [4] and SpacePI [22] the entities involved are modeled as spheres situated in space. SpacePI [22] proposes an extension of the π calculus equipped with time and space. In this algebra processes can communicate if they are sufficiently close, but no shapes are considered. However, in biochemistry, the shape of an enzyme plays a very important role in biochemical interactions. The behaviour or the function of an enzyme is mostly determined by its 3D structure (shape).

Many particle-based approaches, such as MCell [28], Smoldyn [3] and ChemCell [25], are derivatives of the Smoluchowski model. This model describes a solution of interacting chemical particles as spheres moving by Brownian motion until two spheres come within a certain distance of each other causing them to react. As a consequence, these approaches can faithfully describe only reaction-diffusion systems where particles are simple spheres and can be moved altogether only in according to Brownian motion laws, differently from Shape Calculus/BioShape, where each shape can be singularly linked to any motion law.

More similarly to Shape Calculus/BioShape, the particle-based and single-scale approaches proposed in [1] and in [8] also allow particle geometric information to be incorporated. In detail, the mesoscopic simulator proposed in [1] represents biological entities as single particles, spheres or cylinders, or as compound objects formed from the two. Every basic particle can have a number of binding sites associated with it. Particles and compound objects diffuse through the simulation volume using a 3D random walk algorithm. Bonds between particles are broken and created as determined by the user-defined rules. A collision detection algorithm establishes whether particles come sufficiently close to allow bond formation.

The stochastic simulator in [8] handles spatial locality, very low particle concentrations and collision between particles using a discrete 3D grid. Particles move within discrete volumes in discrete time steps. An integer-addressed 3D grid avoids floating-point computation and distance calculations for enabling highly parallel, large-scale simulations using custom hardware.

7 Conclusions and Future Work

We have defined a Shape Calculus that takes into account space, time, shapes, movement and collisions among shapes. The features and the capabilities of modelling of the calculus have been presented in an introductory way, with a running example taken in the biochemical reaction field. Another paper is going to appear from the same authors that introduces the full formal timed operational semantics of the Shape Calculus and a well-formedness result about possible evolutions of network of 3D processes. As future work we intend to study the possibility to provide qualitative and quantitative verification tools for the Shape Calculus. This can be done by applying both abstractions to some parts of the very large semantics of a network of 3D processes and by specifying more concrete information to other parts that are specified at a very high level of abstraction, like the steer function.

References

[1] Meredys (`www.ebi.ac.uk/compneur-srv/meredys.html`).

[2] RCSB - Protein Data Bank (`http://www.rcsb.org`).

[3] S. S. Andrews and D. Bray. Stochastic simulation of chemical reactions with spatial resolution and single molecule detail. *Phys. Biol.*, 1(3–4):137–151, 2004.

[4] R. Barbuti, A. Maggiolo-Schettini, P. Milazzo, and G. Pardini. Spatial Calculus of Looping Sequences. *Electronic Notes in Theoretical Computer Science*, 229(1):21–39, 2009.

[5] E. Bartocci, F. Corradini, M. R. Di Berardini, E. Merelli, and L. Tesei. Shape Calculus. A spatial calculus for 3D colliding shapes. Technical Report 6, Department of Mathematics and Computer Science, University of Camerino, Jan 2010. Available at `http://s1report.cs.unicam.it/6/`.

[6] E. Bartocci, M. R. Di Berardini, F. Corradini, M. Emanuela, and L. Tesei. A Shape Calculus for Biological Processes. In *Proceedings of 11th Italian Conference on Theroretical Computer Science (ICTCS'09)*, pages 30–33, 2009.

[7] L. Bortolussi and A. Policriti. Stochastic concurrent constraint programming and differential equations. *Electronic Notes in Theoretical Computer Science*, 190(3):27–42, 2007.

[8] L. Boulianne, S. Assaad, M. Dumontier, and W. Gross. GridCell: a stochastic particle-based biological system simulator. *BMC Systems Biology*, 2:66, 2008.

[9] F. Buti, D. Cacciagrano, F. Corradini, E. Merelli, M. Pani, and L. Tesei. Bone remodelling in BioShape. In *Proceedings of Interactions between Computer Science and Biology, 1st International Workshop (CS2BIO'10)*, 2010. Available at `http://cosy.cs.unicam.it/bioshape/cs2bio2010.pdf`.

[10] F. Buti, D. Cacciagrano, F. Corradini, E. Merelli, and L. Tesei. BioShape: a spatial shape-based scale-independent simulation environment for biological systems. In *Proceedings of Simulation of Multiphysics Multiscale Systems, 7th International Workshop (ICCS 2010)*, 2010. Available at `http://cosy.cs.unicam.it/bioshape/iccs2010.pdf`.

[11] D. Cacciagrano, F. Corradini, and M. Merelli. Bone remodelling: a Complex Automata-based model running in BioShape. In *Proceedings of Cellular Automata for Research and Industry, 9th International Conference (ACRI'10)*, 2010. Available at `http://cosy.cs.unicam.it/bioshape/acri2010.pdf`.

[12] N. Cannata, F. Corradini, E. Merelli, and L. Tesei. A spatial model and simulator for metabolic pathways. In *Proceedings of Bioinformatics Methods for Biomedical Complex System Applications (NETTAB'08)*, pages 40–42, 2008.

[13] N. Cannata, F. Corradini, E. Merelli, and L. Tesei. A spatial simulator for metabolic pathways. In *Proceedings of MultiAgent Systems&Bioinformatics (MAS&BIO'08)*, pages 31–46, 2008.

[14] L. Cardelli. Brane calculi. In *Computational Methods in Systems Biology, International Conference (CMSB'04)*, volume 3082 of *Lecture Notes in Computer Science*, pages 257–278, 2005.

[15] F. Ciocchetta and J. Hillston. Bio-PEPA: An extension of the process algebra PEPA for biochemical networks. *Electronic Notes in Theoretical Computer Science*, 194(3):103–117, 2008.

[16] F. Corradini and E. Merelli. Hermes: agent-base middleware for mobile computing. In *Formal Methods for Mobile Computing, 5th International School on Formal Methods for the Design of Computer, Communication, and Software Systems (SFM-Moby'05)*, volume 3465 of *Lecture Notes in Computer Science*, pages 234–270, 2005.

[17] C. Ericson. *Real-time collision detection*. Elsevier North-Holland, Inc., 2005.

[18] A. Finkelstein, J. Hetherington, L. Li, O. Margoninski, P. Saffrey, R. Seymour, and A. Warner. Computational challenges of systems biology. *IEEE Computer*, 37(5):26–33, 2004.

[19] D. T. Gillespie. Simulation methods in systems biology. In *Formal Methods for Computational Systems Biology (SFM'08)*, pages 125–167.

[20] C. Hecker. Physics, part 3: Collision response. *Game Developer Magazine*, pages 11–18, 1997.

[21] P. Hunter, W. Li, A. McCulloch, and D. Noble. Multiscale modeling: Physiome project standards, tools, and databases. *Computer*, 39(11):48–54, 2006.

[22] M. John, R. Ewald, and A. Uhrmacher. A spatial extension to the π calculus. *Electronic Notes in Theoretical Computer Science*, 194(3):133–148, 2008.

[23] M. C. Lin and S. Gottschalk. Collision detection between geometric models: A survey. In *Proceedings of IMA Conference on Mathematics of Surfaces*, pages 37–56, 1998.

[24] R. Milner. *Communication and concurrency.* Prentice-Hall, Inc. Upper Saddle River, NJ, USA, 1989.

[25] S. J. Plimpton and A. Slepoy. Microbial cell modeling via reacting diffusive particles. *Journal of Physics: Conference Series*, 16(1):305–309, 2005.

[26] C. Priami and P. Quaglia. Beta binders for biological interactions. In *Computational Methods in Systems Biology (CMSB'04)*, pages 20–33, 2004.

[27] A. Regev, E. Panina, W. Silverman, L. Cardelli, and E. Shapiro. Bioambients: an abstraction for biological compartments. *Theoretical Computer Science*, 325(1):141–167, 2004.

[28] J. R. Stiles and T. M. Bartol. Monte Carlo methods for simulating realistic synaptic microphysiology using MCell. In E. D. Schutter, editor, *Computational Neuroscience: Realistic Modeling for Experimentalists*, pages 87–127. CRC Press, 2001.

[29] K. Takahashi, S. Arjunan, and M. Tomita. Space in systems biology of signaling pathways–towards intracellular molecular crowding in silico. *FEBS letters*, 579(8):1783–1788, 2005.

[30] BioShape. (`http://cosy.cs.unicam.it/bioshape/`).

[31] W. Yi. Real-time behaviour of asynchronous agents. In *Proceedings of CONCUR '90*, volume 458 of *Lecture Notes in Computer Science*, pages 502–520, 1990.