Least Rattling in Smarticles

author list

April 6, 2019

Contents

- Introduction
- regularization in smarticles
- Effective noise Teff, correlation
- Other configurations: random, phase-offset
- breaking this - inertial (do exp't with less inertia?)

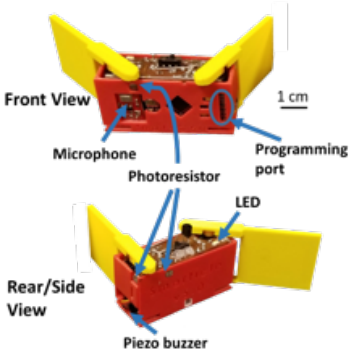


Figure 1: *[label base]* single smarticle with various sensors labelled

- RWRM – graph and diffusion
  as max-caliber
- Discussion

1 Introduction

2 Smarticles

We conducted our experiments with a collection of small simple robots called "Smarticles" (smart active particles) (fig. 1). Each one is comprised of three connected links, with the two hinges being controlled by a programmable micro-controller via stepper motors. When the Smarticle is sitting on its base, the arms do not touch the ground, and so an individual smarticle cannot move. A group of them, on the other hand, can achieve complex motion by pushing off each other. Here we will focus on the regime where their arms are executing a periodic motion pattern, which we call "gait," with no feedback from the environment (note that the gait timing is kept by the controller, and hence the cycle's phase is not affected by forces on the arms). Such gaits can be represented (uniquely up to rate) as closed loops in the two-dimensional conformation space (the two hinge angles) of the Smarticle (fig. 1). In the first part of the paper, we will have all Smarticles execute a synchronized "square" gait (fig. 1). *[Include all the technical details of realization and construction of Smarticles and synchronization in an appendix?]*

We place three Smarticles on a flat aluminium plate, inside a glued-down plastic ring (fig. 2). By moving their arms, the robots can push off each other, while the ring prevents them from decoupling – thus forming an optimal play-ground to look for dynamical emergence. To get data, we use Optitrack *[details]* to track the full time-evolution of the 2D coordinates and body-angle (x, y, θ) of each robot at 120 frames per second (we do not track the arm positions).

For experiment prototyping and better sampling statistics, we have also built a simulation of this setup, described in Appendix B, and benchmarked it against experiments. Both experimental and simulation data are then fed through a pre-processing pipe-line, which mods out the rotation and permutation symmetries of the 3-Smarticle setup, thus reducing the effective configuration space substantially (Appendix C).

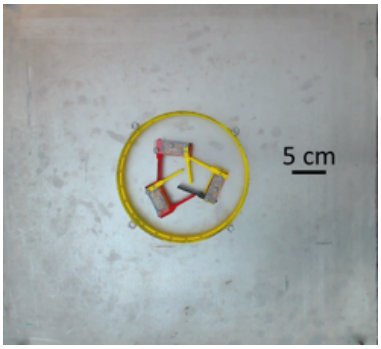


Figure 2: View from above. Regular state testing setup. The whole setup placed on an aluminum plate leveled to $< 0.1°$. The ring is 19.2 cm in diameter, and is glued to the plate below such that it cannot move or deform. The smarticles perform a square gait locked in phase with each other.

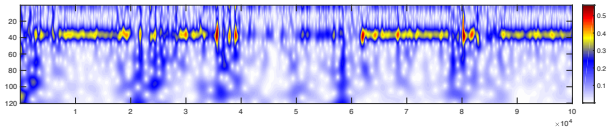


Figure 3: Wavelet transform of dynamics, simulation

3 Regularization

In general, the response properties of a dynamical system to an external drive will depend on the system's configuration. In particular, a given drive can cause chaotic motion in some parts of the configuration space, while being perfectly organized elsewhere. To check if this applies here, we set all three Smarticles to execute a phase-synchronized square gait inside a glued-down ring, track the trajectories, and plot out the wavelet transform of the orientation of any one Smarticle – fig. 3. As the energy from the drive is injected at a particular frequency, we always see a peak in the power-spectrum there. On the other hand, the amount to which that energy gets dispersed to other frequencies before dissipating out varies as the system explores different configurations over time. In the regions where a large broad-band component emerges, we have very noisy dynamics. The lack of a broad-band component, on the other hand, indicates that the dynamics are orderly, which we may be surprised to find in such a messy system.

Taking a closer look at these regular dynamics, we see that there the three Smarticles organize into one of several periodic dynamical patterns (a sort of collective organized dance) *[videos in supplement]*. Figure 4 gives some illustration of the fine-tuning associated with this *[better visualization of regular states???]*. Such self-organization is perplexing, and its interpretation depends on the perspective we take – that of dynamical systems theory, or thermodynamics. *[not synchronization]*

From the context of dynamical systems, perhaps it is not surprising that the dynamics of a time-periodic system are governed by several limit-cycle attractors. On the other hand, not all periodic systems relax to such attractors – even here we can make slight changes to our setup, such as changing relative gait phases, or making Smarticles inertial *[figures for this here?]*, so as to make the system behave chaotically. Furthermore, this system cannot really be seen as deterministic, as we know that it is strongly affected by the stochasticity coming from the hard-boundary interactions between non-smooth shapes, and highly surface-sensitive effects of sliding friction.

From the perspective of equilibrium thermodynamics, where entropy and disorder tend to a maximum, this sort of self-organization may seem extremely surprising. Of course, this system is

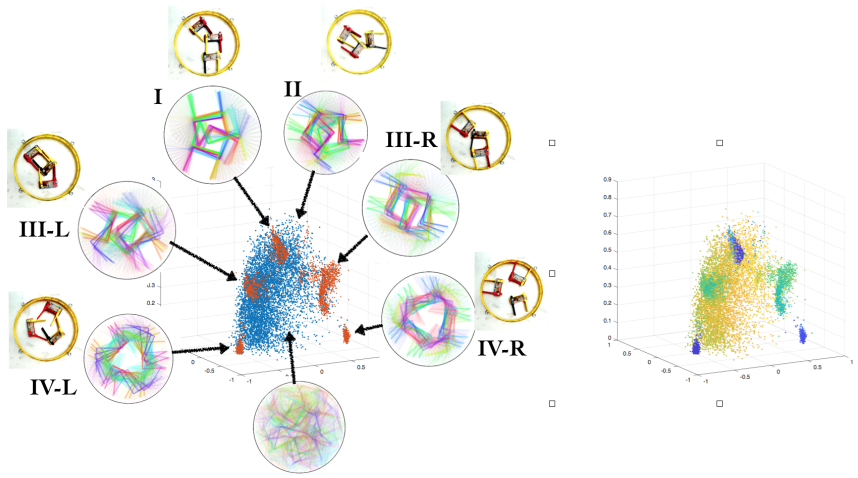


Figure 4: Stroboscopic, 3D projection resolving clusters, of some function of the 9D space (see C), ordered v null, color by delta over one period *[use exp't data for ordered, and simulation for null] [regular states for different drives on same axes]*

far from equilibrium, and so entropy need not grow. Indeed, for a Brownian particle diffusing in a compact domain of inhomogeneous temperature $T(\vec{x})$ (and no other forces), we know that the steady-state probability distribution will be $p_{s.s.} \propto 1/T(\vec{x})$ – which can be far from the maximum-entropy uniform distribution. We want to use this example to motivate our explanation of the fine-tuning we observe.

Indeed, the intuition is the same: probability concentrates in regular states, which are, in some sense, less noisy than the chaotic behavior in other parts of the configuration space. We can make this intuition precise by defining an effective temperature $T_{eff}(\vec{x})$ at every point in the configuration space, and checking if it correlates with $p_{s.s.}(\vec{x})$. While this system is too complicated to calculate $T_{eff}(\vec{x})$ from first principles, we can "measure" it numerically from ensembles of short trajectories starting at $\vec{x}$. Our hypothesis, therefore, is that we can predict the long-term behavior from looking only at short trajectories – which need not be possible for far-from-equilibrium systems.

4 Defining T_{eff}

The question, therefore, is how can we extract a measure of effective noise amplitude from looking at short trajectories. On one hand this system is periodic in time, so one natural metric would be to look at the distance it moves in configuration space over one period. Another natural metric could be inspired by Brownian dynamics $\dot{x} = \sqrt{2\,T(x)} \cdot \xi$, and so $T(x) = \int dt\ \langle \dot{x}(t)\, \dot{x}(0) \rangle /2$, or in higher dimension, we can take the trace of the correlator $T(x) = \sum_i \langle \dot{x}_i\, \dot{x}_i \rangle /2$ (see Appendix C.4 for details). We then color the points in fig. 4 by this T_{eff}, and see that the regions of low T_{eff} correspond precisely to the concentration of steady-state probability. More precisely, fig. *[fig:corr]* shows the scatter plot of $p_{s.s.}(x)$ vs. $T_{eff}(x)$, showing the correlation.

Note that this may at first seem tautological: at long times the system tends to be found in places from where it does not leave. However, below we will explain why this is both, highly non-trivial in a precise mathematical sense, and also profound for understanding real-world strongly-driven systems

5 Drive-specificity

Why should a driven dynamical system get an inhomogeneous $T_{eff}(x)$ landscape? For a given form of the drive signal, different system configurations will have a different response. In particular, some configurations will respond in a more predictable way than others. In our Smarticle setup, for some initial conditions the sources of stochasticity (such as corner collisions) will be more prevalent than elsewhere. This way, a system settles into drive-specific configurations that give more orderly and stable response properties, and changing the drive will change these selected states. For Smarticles, this would imply that the regular states found at long times are specific to the gait, which is confirmed in experiments (fig. *[different gait reg states]*). To illustrate that our approach generalizes beyond periodically-driven systems, we can even make the gait random in time, though identical across Smarticles to allow diversity of response properties *[figure]*.

Another prediction we can make from our framework is that raising the apparent noise uniformly for all configurations $T_{eff}(\vec{x}) \to T_{eff}(\vec{x}) + T_0$ will effectively level the steady-state distribution for a large enough T_0, and thus break the regularization we observe. Experimentally we can realize this by putting smarticles on top of a bed of small, light beads (fig. *[smarticles on beads]*), thus effectively reducing the friction between Smarticles and the table. This makes every interaction have a larger and less predictable effect, which affects all states in a very similar manner.

[Random correlated arms]
[example with more Smarticles]
[inertial smarticles]
[drive-specificity: change drive only and show T(x) change]

6 Broader impact

Our $T_{eff}(\vec{x})$ is effectively a measure of instability of the configuration $\vec{x}$. Hence, a statement that at steady-state the systems tends to be found in configurations that are more stable (in places it does not leave) may seem tautological at first. However, the key here is that this measure is local – depends only on the state $\vec{x}$ itself, and not on the rest of the landscape. Landauer's "blowtorch theorem," *[cite]* on the other hand, shows that for general nonequilibrium systems, the value of the steady-state probability at a point $\vec{x}$ can depend on system properties at any arbitrarily far-away points. Hence having any local quantity that is predictive of the $p_{s.s.}$ is highly non-trivial and is indicative of additional structure in the system. It can, in some sense, be seen as a generalization of equilibrium thermodynamics – there, detailed balance ensures that the steady-state is determined locally. In our case, on the other hand, we try to leverage the additional assumption of "complexity": we think of our system as so complex, high-dimensional, and "generic" that it can essentially be approximated as random. We argue that this is really the "typical" scenario – while it can be violated, such a violation would require fine-tuning, which becomes progressively less likely in more complex high-dimensional systems.

There are several ways to make this discussion more precise. One way to look at it is to consider a Markov process $\dot{p}_i = \sum_j R_{ji}\, p_j - \sum_j R_{ij}\, p_i$, with N states $i \in \{1, ..., N\}$. In general the steady-state distribution $p_{s.s.}(i)$ will depend on all the N^2 transition rates. Now, if we let the transition rates R_{ij} to be independent identically distributed random variables (with some mean $\bar{R}$ and standard deviation σ), and let $N \to \infty$, then it is easy to show *[Appx or citation]* that $p_{s.s.}(i) = \frac{1}{N} + \sum_j (R_{ji} - R_{ij})/(N^2\, \bar{R}) + O(N^{-3})$. The first term $\sum_j R_{ji}$ is the total entrance rate into state i, and may in a real system be hard to measure, as it would require initializing the system in all possible configurations and seeing how often it enters i. The second term $\sum_j R_{ij}$, on the other

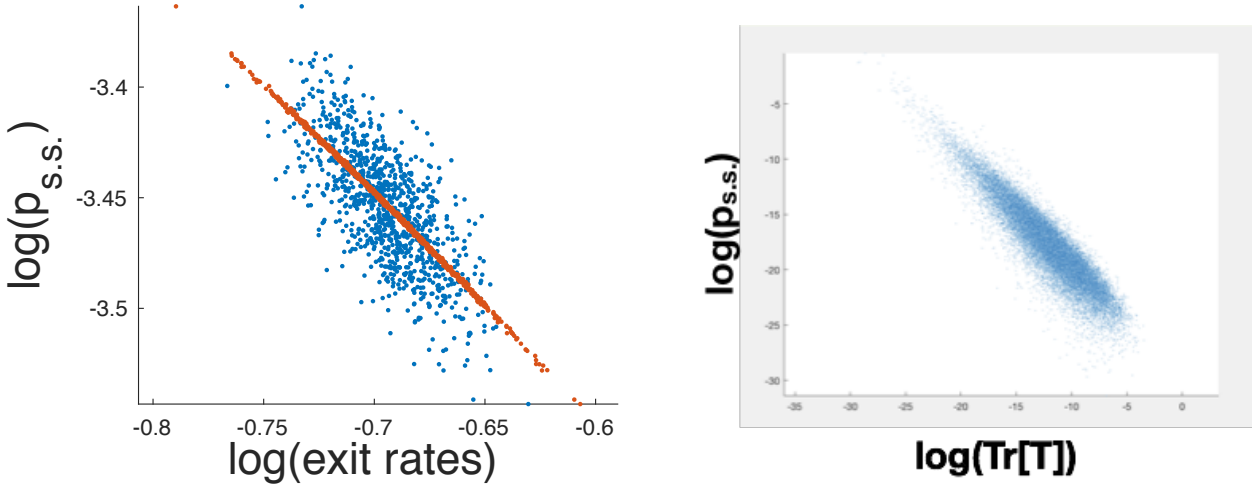


Figure 5: a. Steady state for a Markov process with 1000×1000 random transition matrix. Blue points show exit rates, red points also include the correction from entrance rates. b. Steady state vs. trace of temperature tensor for multivariate inhomogeneous diffusion in a random temperature landscape

hand, is the total exit rate, and is akin to our measure of T_{eff} – it is basically a measure of how stable the state i is and only requires local measurements initialized in that state. We can rewrite the expression for the steady-state focusing on the exit rates: $p_{s.s.}(i) = \frac{\bar{R}}{\sum_j R_{ij}} + \frac{\sigma}{\bar{R}\,N^{3/2}}\,\xi_i$, where $\xi_i \equiv \sum_j \left(R_{ji} - \bar{R}\right) / \left(\sigma\sqrt{N}\right) \sim \mathcal{N}[0,1]$ – univariate Gaussian random variable (note however that it is correlated with $p_{s.s.}(i)$). So while in general we need all N^2 rates to predict the steady-state, the randomness in rates allows us to express it in terms of just N local measurements of the exit rates, up to a noise term from the unobserved entrance rates 5a.

It turns out we can make an even stronger statement if we consider the d-dimensional diffusion process on a compact domain $\dot{x}_i = D_{ij}(x) \cdot \xi_j$ [Ito], $i \in \{1, ..., d\}$, $\vec{x} \in \mathbb{D}^d$, where the diffusion tensor D_{ij} varies smoothly over x, but is otherwise random. Again, as this is a non-equilibrium process, in general the steady-state density at any point will depend globally on the diffusion tensor everywhere: $p_{s.s.}(x) = f_x\left[D_{ij}(y) \mid y \in \mathbb{D}^d\right]$. However, the random choice of $D_{ij}(x)$ simplifies this result: we can numerically show that $p_{s.s.}(x) \sim 1/\mathrm{Tr}\,(T(x))$ for $T \equiv D\,D^T$ 5b, as long as the variation of $\mathrm{Tr}\,(T(x))$ is sufficiently large.

Finally, we can also frame our results in a somewhat different light. Thus, while the exact dynamics generally give a complicated $p_{s.s.}(x)$, we can view $1/T_{eff}(x)$ as the least-informative approximation based on only local information. We think of approximating the system behavior by "typical" dynamics that match the local fluctuation amplitude of the system $T_{eff}(x)$. Concretely, we find the maximum-entropy (or "max-caliber" *[cite]*) probability distribution over trajectories $P[\vec{x}(t)]$, constraining the local velocity fluctuations: maximize

$$\mathcal{S} = P[x]\ \log\left(P[x]\right) + \lambda_0\left(\int \mathcal{D}x\ P[x] - 1\right) + \int dX\ \lambda(X)\left(\sum_i \langle \dot{x}_i\,\dot{x}_i\rangle_X - T_{eff}(X)\right)$$

, where $\langle \dot{x}_i\,\dot{x}_i\rangle_X \equiv \int \mathcal{D}x\ P[x]\ \int dt\ \dot{x}_i(t)\dot{x}_i(t)\,\delta(x(t)-X)$. This gives $P[x(t)] \propto \exp\left[-\sum_i \int dt\ \lambda(x(t))\,\dot{x}_i(t)\dot{x}_i(t)\right]$, which corresponds *[uniquely]* to the stochastic process $\dot{x}_i = \sqrt{2\,T_{eff}(x)} \cdot \xi_i$, whose steady-state dis-

tribution is $p_{s.s.}(x) \propto 1/T_{eff}(x)$. *[explain this paragraph better - too dense, and interesting] [coordinate dependence]*

A Experiment details

Talk about Smarticle synchronization issue and experimental design decisions.

B Simulation details

For easier exploration of hypothesis space of this system, we have constructed a numerical simulation with the following algorithm. We approximate smarticles by three-segment lines (insets in fig. 7b). At each time-step, the algorithm moves the arms slightly according to the chosen gait, and then iteratively cycles through smarticle pairs in random order, checking for collisions, and moving one in each pair slightly according to the net interaction force. If there are multiple points of contact for a given pair, the move is a translation in the direction of the total force, otherwise, it is a rotation about a pivot point chosen so as to balance the forces and torques. Choosing which in the pair moves is random, weighted by their relative friction coefficients (as motivated by difficulty of predicting static friction). Note that since a move can create new collisions with other smarticles, it is important to take small steps and iterate. The algorithm continues looping through pairs until all collisions are resolved, then proceeding to the next time-step. While this describes the core of the algorithm, there are a number of bells and whistles necessary to improve its stability and reliability:

- If two arms are near-parallel when they approach each other, they can pass through each other between ticks without ever intersecting. To prevent this, along with collision detection, we must explicitly test for this in each pair. We then store the order of the arms for a few ticks into the future to prevent them passing through each other in any of those times.

- In case a smarticle with small friction gets trapped between two others, it might rattle back and forth on each iteration of collision-resolution, with no net effect. To prevent this, we temporarily (until next tick) increase its friction each time a smarticle moves.

- In experiment, when resolving collision is too hard, the motor simply does not move (jams up). To allow for that possibility in simulation, we add an exit condition from the collision-resolving loop for when any one smarticle's temporary friction (from last bullet) becomes very large - as this serves as a proxy for how much force a motor must provide. We then move the most colliding arm back to its last time-step, and try collision-resolving again. If everything resolves, that arm will then have a chance to catch up to where it needed to be over the following ticks (its speed being capped at some ω_{max}).

- The resulting simulation turns out to be too clean, despite the multiple stochastic elements of the algorithm, and so dynamical phases much more stable than in experiments. Since we want to study transition statistics, we must find ways to destabilize them. There are a number of places we can add more noise: slight fluctuations of smarticles' positions and angles at each tick, or proportional to each move, randomly varying gait amplitude, varying arm velocity, etc. Adding inertia or inter-Smarticle friction forces (see below) also helps to destabilize the dynamics.

- The ring boundary is implemented similar to other smarticles, and collisions with it are resolved in the same loop. Alternatively, we can implement weakly confining potential to keep smarticles together in a more smooth way (for different experiments).
- It is easy to adjust the simulation to give the Smarticle inertia: at each tick, simply move them according to last step's velocity.
- If inter-Smarticle friction is 0, then each interaction force is directed normal to one Smarticle's surface. To include effects of such friction, we can add a small lateral component to these forces according to the interaction angle.

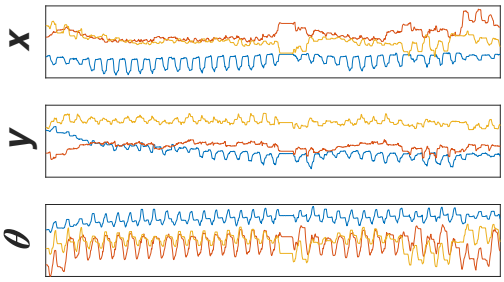
Even with all these additions, many differences remain between simulation and experiment: smarticles have non-zero thickness in experiments, friction forces between smarticles are often important on collisions, there are relief features on smarticle body not present in simulation that can get caught, etc. The consequence in this system is that while qualitative features can be recovered in the simulation, we don't generally expect quantitative agreement. We guess that these differences may become less important for ensembles with more smarticles – as universal collective properties start to dominate. We also guess that analytical predictive power of least-rattling, or other similar techniques, could be more applicable in that regime.

C Data analysis

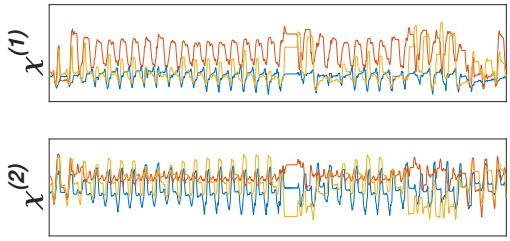
Once we have the tracked coordinates of the smarticles, we want to map the data into a "collective behavior space" – something that can usefully inform us about the dynamical patterns of motion, especially those relevant for slow time-scales. This behavior space will typically be very high-dimensional, and to work with it we try to either cluster the data into different discrete behaviors (or dynamical phases), or perform a dimensionality-reduction to embed the data in a 2 or 3 dimensional space we can visualize (and could subsequently cluster if desired). In either case, what we care about is not the location of points in behavior space (which would be arbitrary anyway), but only about the distances between them – i.e., some measures of similarity of different behaviors.

The raw data from particle tracking gives position and orientation time-traces $(x_i(t), y_i(t), \theta_i(t))$ for each smarticle (fig. 6a), and contains lots of information that is irrelevant for the questions we care about. The first challenge in dealing with this is to chose a set of macroscopic observables of interest (such as collective angular velocity, or average pressure on the walls perhaps). We can then study the well-posed question of how much the different pieces of measured data affect these observables. In principle, understanding this would allow us to define a distance metric on our data space that we could then use for clustering or dimensionality reduction. There are some pieces of data which we know, from first principles, to be entirely irrelevant – corresponding to symmetries in the system. In the simplest setup, these are global rotation symmetry, and smarticle permutation symmetry. Thus, two configurations that differ only by a permutation of smarticles should be considered to have distance =0. While these exact symmetries show us which data is entirely irrelevant, the biggest challenge is in identifying the relative importance of the remaining data for our questions.

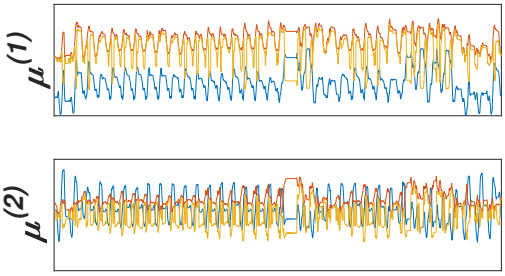
Doing this exactly is an insurmountable feat, and so we must resort to reasonable approximations. While constructing such approximations is in general a rich and interesting problem, here we simply use a reasonable heuristic based on some physical intuitions.



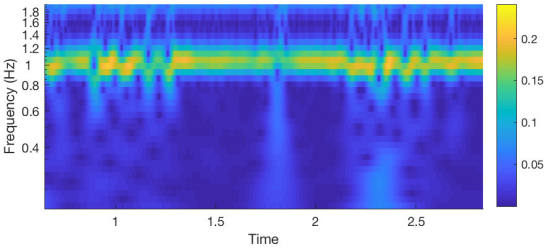
(a) Smarticle coordinates and angles as tracked from experiments



(b) Rotationally invariant observables: c.o.m. coordinate as seen in the reference frame of each smarticle



(c) Permutation invariant observables: first 3 moments of the distributions in 6b



(d) Wavelet transform of one of the curves in 6c. We choose the values along the bright line for our feature vector.

Figure 6: Data processing steps

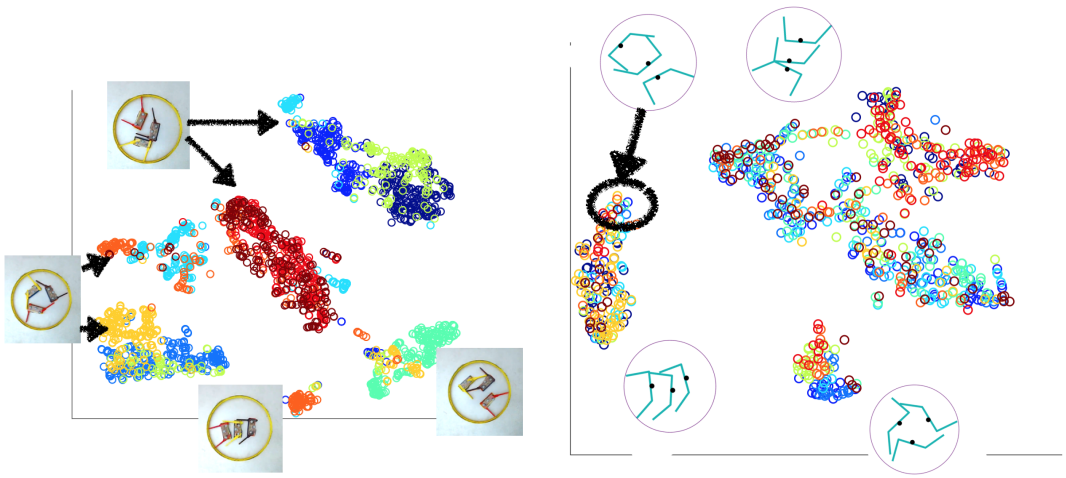
C.1 Modding out symmetries

We begin by constructing functions of the raw data that are invariant under the symmetries we mentioned, but still sensitive to the information we may care about. The two symmetries are global rotations and permutation. To take care of the first, we find the coordinates of the c.o.m. $\vec{X} = \frac{1}{3}\sum_i \vec{x}_i$ (note: permutation invariant) in the reference frame of each smarticle: $\vec{\chi}_i \equiv \mathcal{R}(\theta_i)\cdot(\vec{X} - \vec{x}_i)$, where $\mathcal{R}(\theta_i)$ is the rotation matrix for i-th smarticle (fig 6b). Note that this is not a perfect choice: it is entirely invariant under c.o.m. translations (which is relevant, but we assume it to be small for a confined system), and smarticle orientation matters more the further it is from c.o.m. (so we must assume that their distance to c.o.m. stays relatively constant). We can often get away with such imperfect choices because we have so much overloaded data for the questions.

Next, the permutation-invariant functions we choose are just the first three moments of the distribution of the three $\{\chi_i\}$ at any fixed time. In order for these functions to have relatively similar sensitivity to changes in raw data, it helps to ensure that they have the same units – by raising them to the appropriate fractional powers. I.e., $\mu_1 = \frac{1}{3}\sum_i \chi_i$, $\mu_2 = \sqrt{\frac{1}{3}\sum_i (\chi_i - \mu_1)^2}$, $\mu_3 = \sqrt[3]{\frac{1}{3}\sum_i (\chi_i - \mu_1)^3}$ (fig. 6c). Note also that up to 3, these are the same as the cumulants – which may be a better alternative for more smarticles.

C.2 Capturing dynamics

Now we can proceed to dealing with the fact that behaviors we are looking for are dynamic states, and thus depend on multiple time-points. We thus want to construct some feature vector that captures the dynamics. The simplest idea of simply stacking μ_i at different time-points on top of



(a) Experiment (b) Simulation

Figure 7: Dimensionally reduced data. Insets show what sort of behavior each cluster corresponds to. Note that the cluster positions in the two plots are unrelated as the 2D embedding was done separately on two data sets. Color indicates different trials of the same experiment – confirming that the behaviors are reproducible.

each other is not obviously wrong, but will give a feature vector that is very sensitive to the noise. Instead, the dynamical features we are looking for are better captured by Fourier modes. Further, since the dynamical phase can change on the time-scale of a few arm oscillations, we don't want to transform the entire time-series, but instead consider a continuous wavelet transform of the data (fig. 3). This beautifully separates out behaviors on different time-scales, and we can choose our feature vector based on the time-scale(s) we most care about. The simplest choice here is to take the time-scale corresponding to one gait-period – since most of the activity happens there. Again, this is throwing away lots of data, but since at the end we just care about some low-dimensional behavior space, we are still left with more than enough (including other time-scales had little effect).

This way our feature vector consists of 6 complex numbers (3 moments for each dimension of the 2D χ vector). We do care about relative phase of oscillation in the wavelet transform – to keep this in the most unbiased way, we simply take the 36 possible differences between the 6 complex numbers, and then take their absolute value. While this makes the feature vector unnecessarily long, this is fine since we only care about the relative distances between these, and duplicate vector components will simply sum up.

C.3 Dimensional reduction and clustering

Now that we have our feature vector, we can dimensionally reduce or cluster the data. The former is, in a sense, a "smoother," less dramatic operation, and thus turns out to produce more accurate results. We can understand this by realizing that in either case, the goal is to preserve pair-wise distances as best as possible, and this is much easier to accomplish if you allow points to be placed anywhere in the $\mathbb{R}^2$ plane, rather than in one of 5 discrete states. Curiously, clustering even works better when performed on dimensionally reduced, rather than original data. The other advantage of dimensionally reducing first is that this gives us more information about how the data is distributed, without imposing the assumption that it is well-clustered.

We use Matlab's built-in t-Distributed Stochastic Neighbor Embedding (tSNE) algorithm, which

basically tries to embed the data in a 2D space preserving all the pair-wise distances as much as possible (fig. 7a). It has the effect of largely preserving, or even highlighting, clusters in the original data, but also allows visualizing transitions between them and other hierarchical structure in the data. Note that the two pairs of clusters corresponding to the same pictures are chiral complements (the other two clusters correspond to non-chiral states). In section **??** below we will cluster this data and study the slow transition statistics between different dynamical phases.

C.4 Defining T_{eff}

D Random dynamical systems

D.1 Random Markov process

D.2 Diffusion in random medium

D.3 Random discrete map

D.4 Max-entropy modelling

References