Dice Simulation

Daniel Chin and Michael Li
Oct. 2019

1. Introduction

In this project, we use the model of rigid body motion to simulate the roll of a dice.

The dice is represented by a uniform cube, with 8 points of interest at its corners.

We focus on investigating the chaotic structure in the initial condition space.

2. Model

2.1. Rigid Body Motion

The rigid body consists of 8 points of interests at the corners of the cube. This writeup refers to them as **nodes**.

The total mass and center of mass of the system, as well as the velocity of the center of mass are defined as follows:

$$M = \sum_{k=1}^{n} M_k\,, \qquad M\overrightarrow{X_{cm}} = \sum_{k=1}^{n} M_k\,\overrightarrow{X_k}, \qquad M\overrightarrow{U_{cm}} = \sum_{k=1}^{n} M_k\,\overrightarrow{U_k}$$

Let $\vec{L}$ be the angular momentum of the system about its center of mass:

$$\begin{aligned}\overrightarrow{L(t)} &= \sum_{k=1}^{n} M_k\left(\overrightarrow{X_k} - \overrightarrow{X_{cm}}\right) \times \overrightarrow{U_k} \\ &= \sum_{k=1}^{n} M_k\left(\overrightarrow{X_k} - \overrightarrow{X_{cm}}\right) \times \left(\overrightarrow{U_k} - \overrightarrow{U_{cm}}\right)\end{aligned}$$

Differentiate both sides with respect to t and use the identity $\left(\overrightarrow{U_k} - \overrightarrow{U_{cm}}\right) \times \left(\overrightarrow{U_k} - \overrightarrow{U_{cm}}\right) = 0$

$$\begin{aligned}\frac{d\vec{L}}{dt} &= \sum_{k=1}^{n}\left(\overrightarrow{X_k} - \overrightarrow{X_{cm}}\right) \times \left(M_k\frac{d\overrightarrow{U_k}}{dt} - M_k\frac{d\overrightarrow{U_{CM}}}{dt}\right) \\ &= \sum_{k=1}^{n}\left(\overrightarrow{X_k} - \overrightarrow{X_{cm}}\right) \times \overrightarrow{F_k} \\ &= \sum_{k=1}^{n}\overrightarrow{\widetilde{X_k}} \times \overrightarrow{F_k} \\ &= \vec{\tau}\end{aligned}$$

where $\overrightarrow{F_k}$ denotes the external force acting on a node, $\overrightarrow{\widetilde{X_k}}$ is the positional vector of this node relative to the center of mass, and $\vec{\tau}$ is the total torque of the system.

$$M\frac{d\overrightarrow{U_{cm}}}{dt} = \vec{F} = \sum_{k=1}^{n}\overrightarrow{F_k}$$

Since we want the motion to be rigid, we should solve for the angular velocity. So, we shall introduce the individual node velocity $\overrightarrow{u_k(t)}$:

$$\overrightarrow{u_k(t)} = \overrightarrow{U_{cm}(t)} + \vec{\Omega} \times \left(\overrightarrow{X_k(t)} - \overrightarrow{X_{cm}(t)}\right)$$

Then we plug the above results into the previous definition of torque:

$$\begin{aligned}\overrightarrow{L(t)} &= \sum_{k=1}^{n} M_k \left(\overrightarrow{X_k(t)} - \overrightarrow{X_{cm}(t)}\right) \times \left(\overrightarrow{\Omega(t)} \times \left(\overrightarrow{X_k(t)} - \overrightarrow{X_{cm}(t)}\right)\right) \\ &= \sum_{k=1}^{n} M_k \left\|\overrightarrow{X_k(t)} - \overrightarrow{X_{cm}(t)}\right\|^2 \overrightarrow{\Omega(t)} - \sum_{k=1}^{n} M_k \left(\overrightarrow{X_k(t)} - \overrightarrow{X_{cm}(t)}\right)^2 \cdot \vec{\Omega} \\ &= I(t)\, \overrightarrow{\Omega(t)}\end{aligned}$$

where $I(t)$ is a 3×3 matrix, denoting the moment of inertia:

$$\boldsymbol{I(t)} = \sum_{\boldsymbol{k=1}}^{\boldsymbol{n}} \boldsymbol{M_k} \left(\left\|\overrightarrow{\boldsymbol{X_k(t)}}\right\|^{\boldsymbol{2}} \boldsymbol{E_{3\times 3}} - \overrightarrow{\boldsymbol{X_k(t)}} \left(\overrightarrow{\boldsymbol{X_k(t)}}\right)^{\boldsymbol{T}}\right) \tag{1}$$

We can solve for $\vec{\Omega}$ by computing

$$\vec{\Omega} = \left(\overrightarrow{I(t)}\right)^{-1} \overrightarrow{L(t)}$$

Once we know $\vec{\Omega}$, we construct the rotation matrix $R(\vec{\Omega}, \Delta t)$. In particular, R rotates every node about the rotating axis for an angle $\overrightarrow{\Omega(t)}\Delta t$.

Then we construct the projection matrix onto the unit vector in the direction of $\vec{\Omega}$, which is rotational-invariant.

$$P(\vec{\Omega}) = \frac{\vec{\Omega}}{\|\vec{\Omega}\|}\left(\frac{\vec{\Omega}}{\|\vec{\Omega}\|}\right)^T$$

We express $\vec{\tilde{X}}$ as the following:

$$\vec{\tilde{X}} = P(\vec{\Omega})\vec{\tilde{X}} + \left(E_{3\times 3} - P(\vec{\Omega})\right)\vec{\tilde{X}}$$

The former ingredient is rotational invariant, while the latter ingredient is rotational about $\|\vec{\Omega}\|\Delta t$ through the origin, normal to $\vec{\Omega}$.

Finally, by taking the orthonormal basis $(\vec{v_1}, \vec{v_2})$, where $\vec{v_1} = \left(E_{3\times 3} - P(\vec{\Omega})\right)\vec{\tilde{X}}, \vec{v_2} = \frac{\vec{\Omega}}{\|\vec{\Omega}\|} \times \vec{v_1}$, we achieve the orthogonal matrix in terms of $\vec{\tilde{X}}$ and $\|\vec{\Omega}\|\Delta t$:

$$R(\vec{\Omega}, \Delta t)\vec{\tilde{X}} = P(\vec{\Omega})\vec{\tilde{X}} + \cos(\|\vec{\Omega}\|\Delta t)\left(E_{3\times 3} - P(\vec{\Omega})\right)\vec{\tilde{X}} + \sin(\|\vec{\Omega}\|\Delta t)\frac{\vec{\Omega}}{\|\vec{\Omega}\|} \times \vec{\tilde{X}}$$

2.2. Gravity

$$G = mg$$

where **gravitational constant** $g = -9.8\, m/s^2 = -980\, cm/s^2$.

2.3. Interaction with the Ground

The ground is modelled as a "very stiff trampoline". In the 3D space $x - y - z$, the ground is always at $z = 0$.

2.3.1. Normal Force

A node receives an upward normal force proportional to how deep it penetrates the ground:

$$F_{normal} = -\min(0, X_z) \cdot k$$

where the **stiffness of ground** $k = 500000\, N/cm$;
X_z is the vertical component of the position of the node.

2.3.2. Damping Force

The collision with the ground should transfer some mechanical energy to heat. To simulate that, we apply a damping force whenever a node is below $z = 0$:

$$\overrightarrow{F_{damping}} = -F_{normal} \cdot U_z \cdot \gamma$$

where the **damping coefficient** $\gamma = 0.005\, s/cm$;
U_z is the vertical component of the velocity of the node.

2.3.3. Friction

$$F_{friction} = -\left(F_{normal} + \left\|\overrightarrow{F_{damping}}\right\|\right) \cdot \mu \cdot \frac{U_{x,y}}{\left\|U_{x,y}\right\|}$$

where the **friction coefficient** $\mu = 1.2$.

We do not model static friction since we end the simulation before the dice fully stabilizes. (see section 2.7)

2.4. Adjusted Moment of Inertia

If we compute the **moment of inertia** I from the 8 nodes via (1), we will overestimate, since a real dice has uniform concentration of mass throughout the cube. Fortunately, we can simply pretend I is a different value without breaking any physics:

$$I = \frac{1}{6} m s^2 \cdot E_{3\times 3}$$

where s is the diameter of the cube.

Since this matrix is invariant to rotation, we do not need to update I during the simulation.

2.5. Other Constants

The radius of the dice is 1 cm.

The mass of the dice is 8 kg. (This does not matter, because mass cancels out in every interaction.)

The initial height of the dice is 15 cm.

The time step $\Delta t = 0.001\, sec$.

2.6. Deviations from Reality

2.6.1. Rigid Body

If you drop a basketball with a backward spin, it is likely to bounce up with a forward spin. That is because the ball stores energy in the form of rotational elastic deformation as it hits the ground. The stored potential energy then spins the ball in the reversed direction when it bounces up.

Since we model the dice as a rigid body, we will lose this effect.

2.6.2. Rigid Ground and No Air Drag

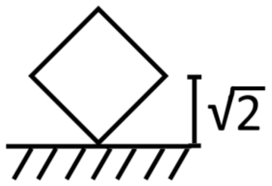
We do not model for ground deformation or air drag.

2.6.3. Infinitely Sharp Corners

A real dice has blunt corners. As a consequence, when it spins around an axis that is not perfectly aligned with its diagonal, the ground contact will involve relative motion. The acceleration this effect has on the dice will give the dice an extra torque.

In our simulation, we model the dice as a perfect cube whose corners are infinitely sharp right angles. Thus, we lose the above extra torque.

2.7. Detecting the End of Simulation and Calling the Outcome



The potential energy of the dice when it is standing on its edge is $mg\sqrt{2}cm \approx 1.1\,J$. This is the critical threshold for the dice to flip over and change face. (Elastic potential energy is neglected here.)

The simulation is ended preemptively whenever the total energy in the system is lower than 80% of the above threshold. This is to reduce simulation time.

To decide which face is facing upwards, the program takes the highest four nodes and lookup which face they describe.

3. Numerical Method

We use forward Euler's method to perform a timestep, given $\overrightarrow{X_{cm}(t)}, \overrightarrow{\widetilde{X_k}(t)}, \overrightarrow{U_{cm}(t)}, \overrightarrow{L(t)}$, and all forces.

We first update the linear motion:

$$\overrightarrow{U_{cm}(t+\Delta t)} = \overrightarrow{U_{cm}(t)} + \frac{\overrightarrow{F(t)}}{m}\Delta t$$

$$\overrightarrow{X_{cm}(t+\Delta t)} = \overrightarrow{X_{cm}(t)} + \overrightarrow{U_{cm}(t)}\Delta t$$

```
total_force = sum(force, 1) + GRAVITY * N_POINTS;
CM_linear_v = CM_linear_v + total_force / N_POINTS * delta_t;
CM_position = CM_position + CM_linear_v * delta_t;
```

Then the rotational motion. We start by updating the angular momentum:

$$\overrightarrow{\tau(t)} = \sum_{k=1}^{n} \overrightarrow{X_k} \times \overrightarrow{F_k}, \qquad \overrightarrow{L(t+\Delta t)} = \overrightarrow{L(t)} + \overrightarrow{\tau(t)}\Delta t$$

```
total_troque = sum(cross(force, X));
L = L + total_troque * delta_t;
```

Then let MATLAB solve for the angular momentum, and populate a transformation matrix $(\Omega\times)$ such that for $\forall\vec{a}$, $(\Omega\times)\cdot\vec{a} = \vec{\Omega}\times\vec{a}$:

$$\vec{\Omega} = I^{-1}L, \qquad \Omega\times = \begin{pmatrix} 0 & -\Omega_z & \Omega_y \\ \Omega_z & 0 & -\Omega_x \\ -\Omega_y & \Omega_x & 0 \end{pmatrix}$$

```
angular_v = I \ L';
angular_v_cross = [
    0               -angular_v(3)   angular_v(2);
    angular_v(3)   0               -angular_v(1);
   -angular_v(2)   angular_v(1)   0             ;
];
```

Given angular velocity and Δt, all nodes rotate by the linear transformation R:

$$p = \frac{\vec{\Omega}\,\vec{\Omega}^T}{\|\vec{\Omega}\|^2}, \qquad \Delta\theta = \|\vec{\Omega}\|\Delta t, \qquad R = p + \cos(\Delta\theta)(E_{3\times3} - p) + \sin(\Delta\theta)\frac{(\Omega\times)}{\|\vec{\Omega}\|}$$

One may be tempted to approximate $\cos(\Delta\theta) \approx 1$, because the error is only in the order of magnitude of Δt^2. However, doing that would deform the rigid body as time goes by, so we must keep the *cos* term.

Finally, update the position of the nodes according to the rotation:

$$\overrightarrow{X_k(t+\Delta t)} = R\cdot\overrightarrow{X_k(t)}$$

```
norm_angular_v = norm(angular_v);
if norm_angular_v ~= 0
    projection_matrix = angular_v * angular_v' / norm_angular_v ^ 2;
    delta_orientation = norm_angular_v * delta_t;
    R = projection_matrix ...
        + cos(delta_orientation) * (eye(3) - projection_matrix) ...
        + sin(delta_orientation) * angular_v_cross / norm_angular_v;
    X = X * R;
end
```

After updating the rigid body, the program evaluates the forces, increments $t = t + \Delta t$, and proceeds to the next timestep.

Notice that in the MATLAB code, the way inner products are written out may look reversed (instead of $A * \vec{b}$ you see $\vec{b} * A$). That is simply because in Math we write vectors as columns, but in MATLAB it is a convention to store vectors as rows so that they are more compact to display on console.

4. Analysis and Experiment Results

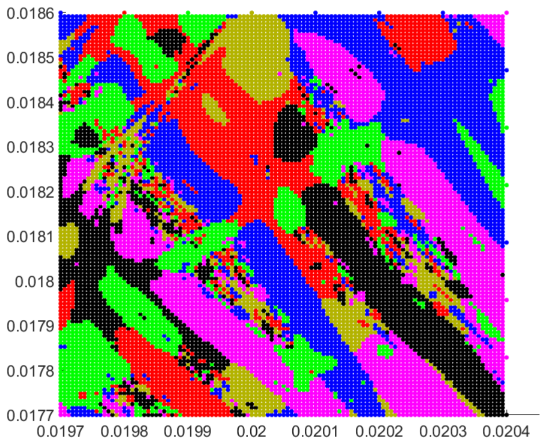
4.1. Cell Structure in Initial Condition Space

Our physics model is deterministic:

$$y = f(ic)$$
$$y \in Z \cap [1, 6], \qquad ic \in IC$$

f is the simulation. y is the **outcome**, the final number given by the roll of a dice. ic is a particular configuration of the **initial condition**. IC is the entire **initial condition space** containing all possible ic.

If we color the initial condition space IC according to outcome y:



We can see compartments. This writeup refers to the compartments as **cells**. Each cell corresponds to a specific y.

The scatter plot on the left is 2D just for visualization. In this project, we consider IC to have 6 dimensions: 3 from initial linear velocity U_{cm} and 3 from initial angular momentum L.[1]

Now, consider this hypothesis:

$$\lim_{k \to 0} f(ic + k \cdot \epsilon) = f(ic) \tag{2}$$

where ϵ is an arbitrary 6-dimensional vector. (2) intuitively describes that you can move and still stay in the same cell as long as your step is arbitrarily small. That makes sense because physics laws are mostly continuous. However, if (2) is true for all points, then apparently the entire IC space can only have one color! That is not right. More specifically, (2) fails to describe the borders in the cell structure of IC.

The borders in IC, as the below video shows, is when the dice ends up standing on its edge.

youtu.be/lm123dN5NC0?t=11s In this video, the x axis is initial condition. The y axis is the time it takes for the dice to settle down, which approaches infinity at the border.

[1] Why not consider initial height h? If h was variable, the research question would have many trivial solutions: set $h = 0$ and let go of the dice. Hence, we decide that h should be a constant.

There exist exact solutions of ic such that the dice will stand on its edge forever. These points in IC have outcome y *undefined*, because they will never give us a valid face result. We are ready to improve (2):

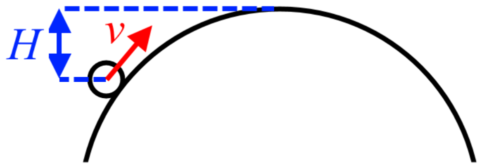
$$\lim_{k\to 0} \boldsymbol{f}(\boldsymbol{ic} + k\boldsymbol{\epsilon}) = \boldsymbol{f}(\boldsymbol{ic}) \text{ where } \boldsymbol{f}(\boldsymbol{ic}) \text{ has definition.} \tag{3}$$

(3) is what we believe to be true. Sadly, we have not been able to prove it. The rest of this writeup will treat (3) as an assumption.

What (3) describes is what we call a **wedged continuous space**. To understand (3) we need to understand what we mean by **wedge**.

4.2. Wedged Continuous Space

Let us take a simpler example. Consider a ball climbing a hill.



The hill is a frictionless arc whose top is H higher relative to the ball. The ball has mass m and starts out with a linear velocity tangent to the hill whose mode is v. No external forces. The outcome of the experiment, y, denotes whether the ball will roll pass the hill or return to where it starts.[2]

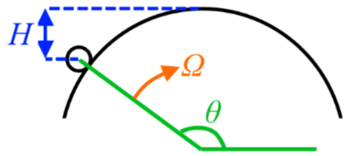
When the initial kinetic energy of the ball equals the potential energy the ball will have gained at the top, $i.e.$ $\frac{1}{2}mv^2 = mgH$, what will the outcome y be?

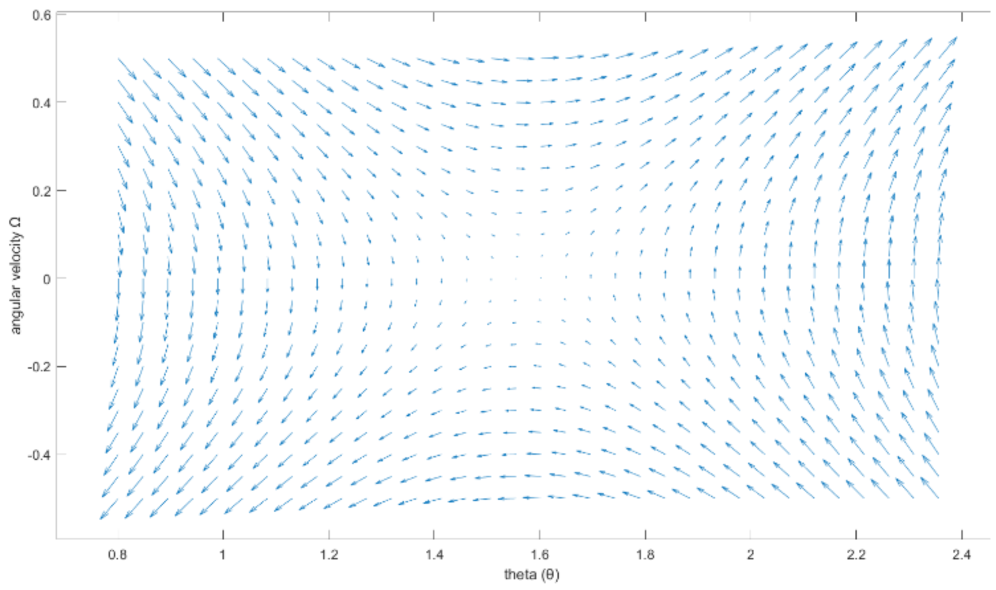
Will the ball roll pass the hill? If so, then the ball must have been at the top of the hill with some non-zero velocity. This breaks the conservation of energy, hence impossible. Will the ball return to where it starts? If so, then the ball's velocity along x axis must go from positive to negative, passing zero velocity at some point, either at the top of the hill or before the top. Can the ball reach zero velocity at the top? If we rewind time from the situation where the ball sits still at the top, we see that the ball will never move, unable to reach its initial condition in finite time, hence impossible. Can the ball reach zero velocity before reaching the top? Again, this breaks the conservation of energy. In conclusion, the ball will neither roll pass the hill nor return to where it starts. Outcome y is undefined!

The ball will approach the top of the hill as $t \to \infty$, but never reaching the top.

[2] For more careful readers, please consider the ball and the hill to be a ring and a rail, because we are assuming the ball never leaves the surface.

We think “physics is continuous”, but y has an undefined point separating two distinct values! This may feel paradoxical. To gain insight, let us borrow a powerful visualization tool, phase space. Let x axis represent θ, and y axis represent angular velocity Ω of the ball in regard to the center of the arc.

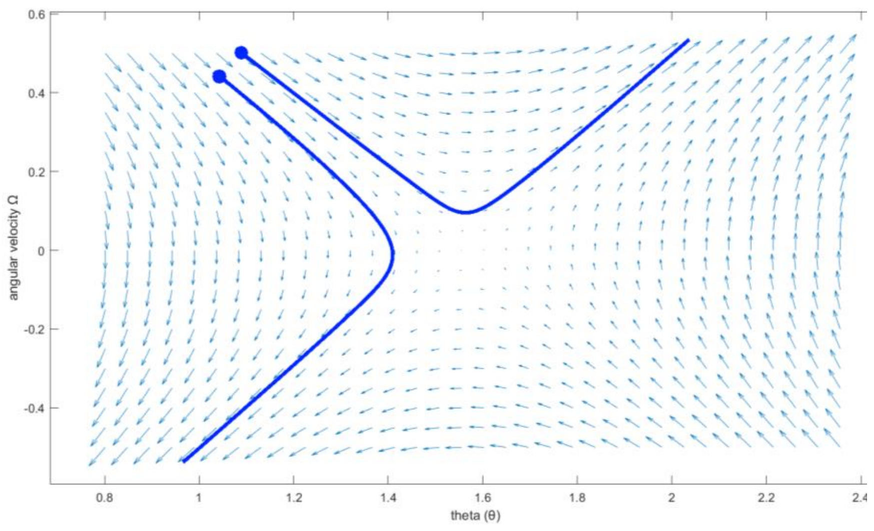




Each point in the phase space represents a state of the ball-hill system. The evolution of the system can be represented by a path in the phase space. For a path to be valid, it must follow the vector field in the image, which is given by:

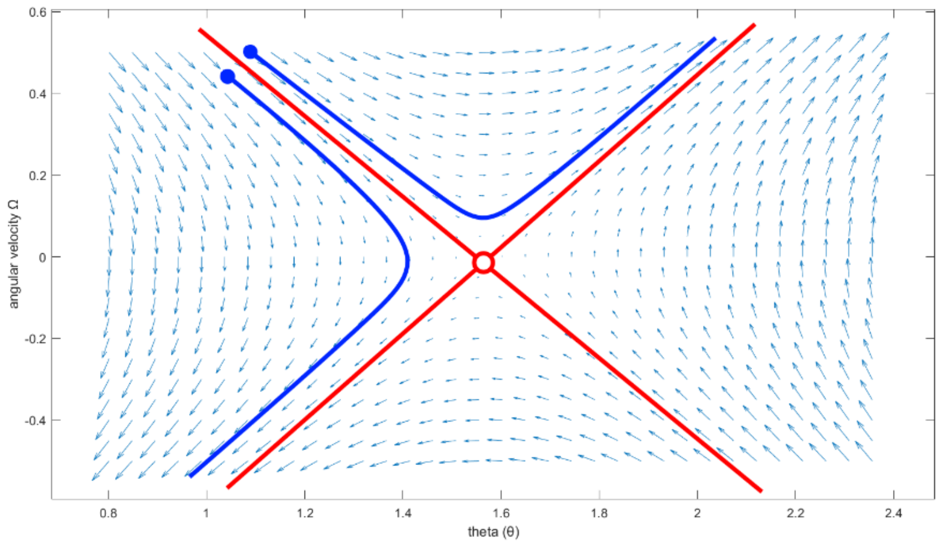
$$\frac{d\theta}{dt} = \Omega, \qquad \frac{d\Omega}{dt} \propto -\cos(\theta)$$

(See the appendix 8.1 for how this is derived.)



Now we can translate our vague feeling “physics is continuous” into something tangible: *the vector field is continuous and differentiable*. Our “paradox”, in the context of phase space, thus translates to: two arbitrarily close initial conditions can lead to separated paths in the continuous and differentiable vector field. How can this happen?

Normalizing the vector field has the effect of ignoring time in the phase space. In the normalized vector field, point $\left(\frac{\pi}{2}, 0\right)$ evaluates to $\frac{\overrightarrow{(0,0)}}{0}$, which is undefined! This point corresponds to an unstable equilibrium of the system. Four paths that connect to $\left(\frac{\pi}{2}, 0\right)$ represent the scenarios when the ball approaches the top of the hill as $t \to \infty$. The four paths separate the phase space into four sections.



Within each section, two arbitrarily close initial conditions always lead to arbitrarily close paths. Only when two initial conditions are in neighboring sections, separated just by the border, can the anomaly appear.

In a wedged continuous space, points like $\left(\frac{\pi}{2}, 0\right)$ are the **wedges**. They divide the phase space into sections in terms of the topology of the evolution of the system. Similarly, the initial condition space IC can also be divided into discrete cells.

The dice simulation is largely similar to the ball-hill experiment, since the trajectory of the center of mass of the dice when it rotates about its edge is a circular arc, just like the ball-hill experiment.[3]

Hence, we claim that (3) and (4) are properties of the initial condition space:

$$\lim_{k\to 0} f(ic + k\epsilon) = f(ic) \text{ where } f(ic) \text{ has definition.} \tag{3}$$

$$f(ic) \text{ is undefined at unstable equilibriums} \tag{4}$$

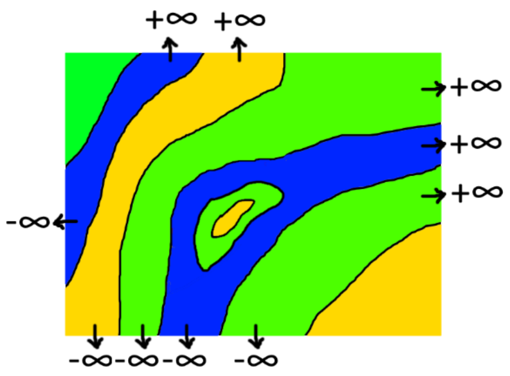
Sadly, we cannot prove (4) either. Keep in mind that (3) and (4) are unproved assumptions for the rest of this writeup.

[3] It is still different from the ball-hill experiment because a) the dice has positive moment of inertia, and b) the pivot edge may change position as it interacts with the ground.

4.3. Redundant Dimensions

Consider the 6-dimentional initial condition space IC^6 described by (3) and (4). It involves a 6D topological cell structure. A vertex in 6D must connect to 7 cells. Each cell corresponds to an outcome y. Assume that two adjacent cells cannot be of the same y. Because there are only 6 faces on a dice, i.e. 6 possible values for y, IC^6 must not have any vertex! That may sound strange, but all it says is that IC^6 has **redundant dimensions**.

Here is an intuitive way to understand redundant dimensions. Imagine I control the initial linear velocity of the dice, and you control the initial angular momentum. No matter how I continuously change the initial linear velocity, can you continuously change the initial angular momentum, so that y always stays constant? That feels possible.



Avoiding vertices in 2D

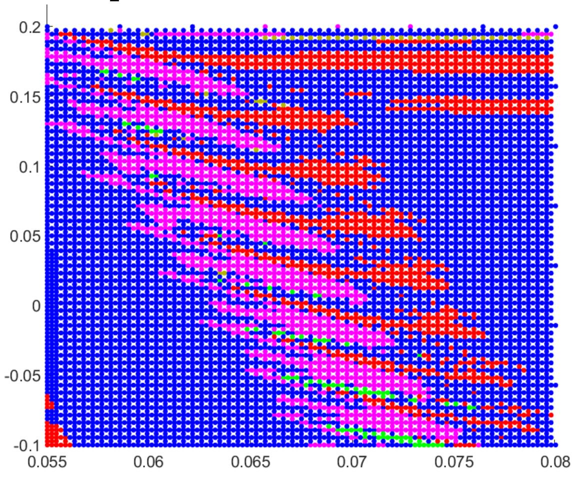
Here is a more formal statement. If there exists a continuous mapping g from IC^n to R^{n-1}, such that all solutions of $g(ic) = x$ leads to the same outcome $f(ic)$, then IC^n has no vertices.

g can be easily constructed if we make the gradient of g perpendicular to the borders between the cells.

It will be interesting to see more rigid analysis on this.

5. Other Experiment Findings

5.1. Repetition

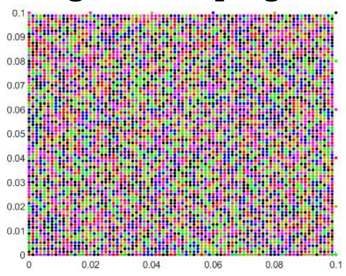


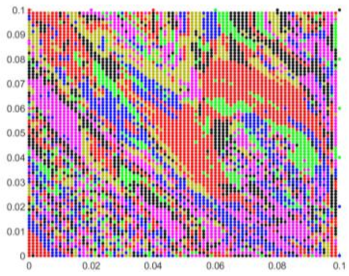
There seems to be a periodic pattern in this section of IC. A valid question is: are these different cells, or many 2D slices of the same 6D cell? Theoretically, programming should be able to answer this question. However, that requires exploring the 6D space to see if the slices are connected. The high dimensionality raises very rapidly the time complexity of the program. So, it is practically highly difficult.

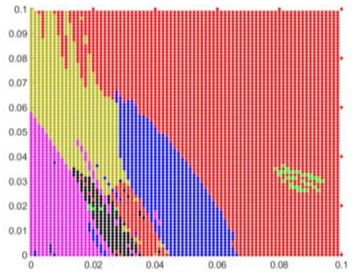
This is a common obstacle for many of our inquiries in this project: high dimensional data are hard to compute, and even harder to visualize.

The cause of this repetition is unknown.

5.2. Higher Damping Coefficient Leads to Less Chaotic IC


$\lambda = .0005$

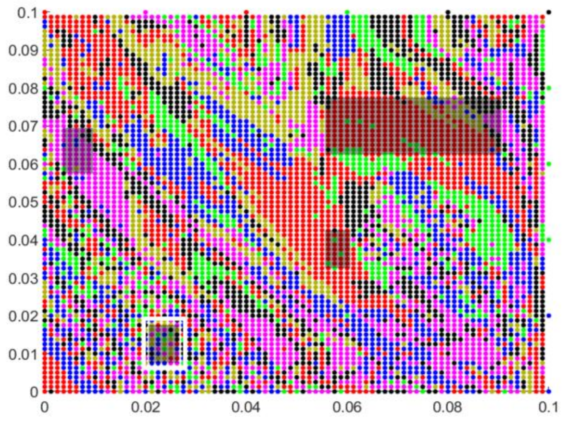

$\lambda = .005$

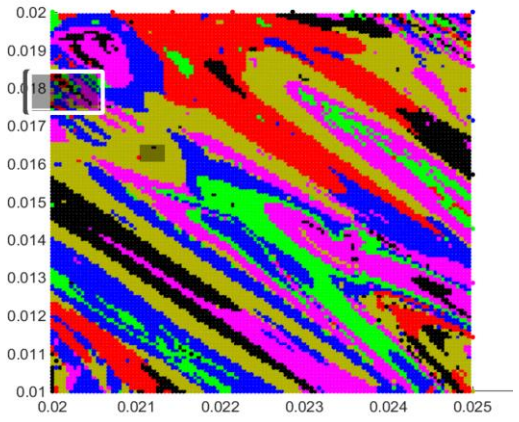

$\lambda = .001$

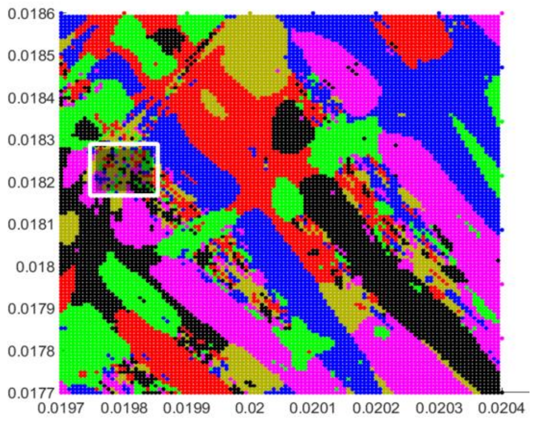
This is likely because a lower damping coefficient λ makes the simulation longer, enabling more events to happen.

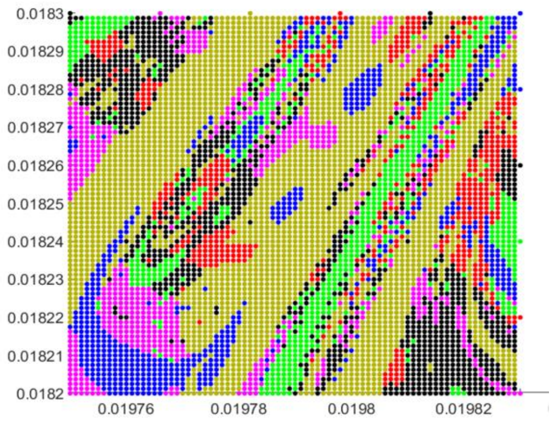
5.3. "Fractal" Complexity

More details emerge in IC as you progressively zoom in, similar to how fractal shapes behave.


1


2


3


4

Each plot magnifies its previous plot. A white box is used to frame the magnified section in the previous plot.

5.4. Vertex Hunt

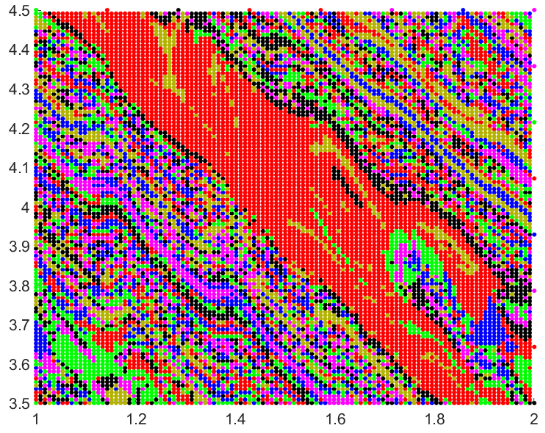
We tried to zoom in and locate a vertex connecting three cells. As we zoom in, the borders become increasingly complex and we soon lost the vertex.

This property of IC may be intrinsic to the physics model, or could be caused by imperfections of the simulation.

To hunt for the vertex yourself or visit the rest of the plotted IC^2, please download our "interactive scatter" at Google Drive: drive.google.com/drive/folders/1CNYjOXlDcMYt4jTBibmj-CmBpp07bfq5?usp=sharing

5.5. Ground Contact Event Sequence

We take a large cell and sample three initial conditions:

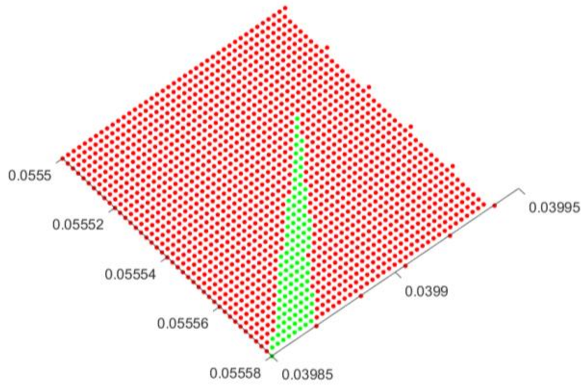


The dice rolling animation can be found at 0:47 of the video demo: youtu.be/lm123dN5NC0?t=47s

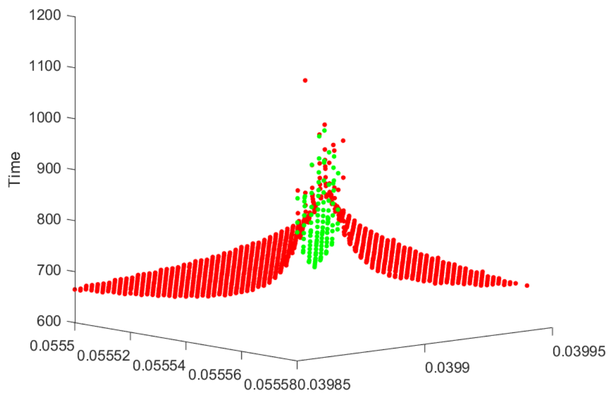
To analyze the "topology" of the throws, we make the program remember the sequence of nodes contacting the ground. We define two ic's to be **topologically the same** if the ground contact event sequence is identical. The three rolls in the video turn out to be topologically different. (Run code topologyOfRedSea.m to see the result)

(IC can thus be further divided into smaller cells according to ground contact event sequence.)

5.6. 2D Scatter with Time Axis



Top view



Angled view

5.7. What it Means to Be in the Same Cell

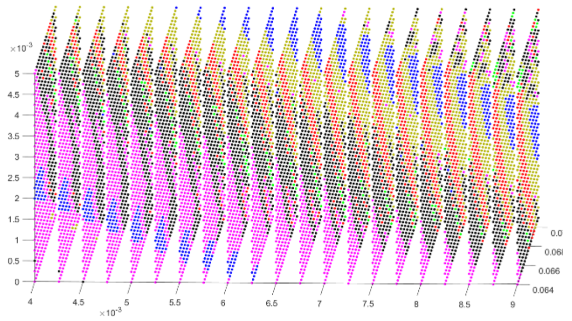
Please find the animation at 0:30 of the demo video: youtu.be/lm123dN5NC0?t=30s

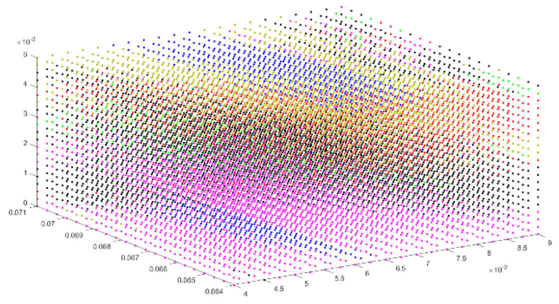
Judging from the dice rolling animation, the three yellow throws look similar. Among the three blue throws, the bottom one goes to "1" via "4", while the other two go to "1" via "5". It would be interesting to see if the two blue "1" – via – "5" sections are fundamentally a same 6D cell.

Note that the yellow ic's are topologically different in terms of ground contact event sequence.

6. Future Work

6.1. Replace 3D Scatter with 2D Animation





In an attempt to visualize IC^3, we tried 3D scatter plot. As the plot rotates, points become a mess and the viewer cannot really see the inside of the 3D area.

Instead of using the 3rd spatial axis, we should have used the time axis, i.e. a video of a 2D scatter, where the third component of ic changes with time.

6.2. Make Base Vectors Perpendicular

When we lower the dimensionality of IC^6 to 2 and 3, we generated the base vectors randomly:

```
L = [x y] * [3.2612 3.1314 0.0606; 2.1595 4.5284 3.9535] + [0 100 0];
CM_linear_v = [x y] * [2.7083 3.4373 4.4365; 2.4650 1.4853 0.6534];
```

This was a mistake. Future analysis should keep the base vectors perpendicular, otherwise the resulting scatter plots may look skewed.

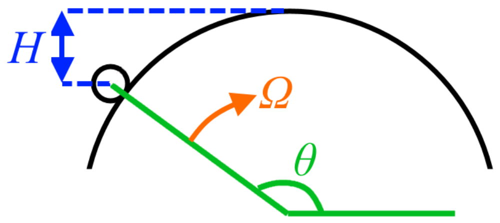
7. Acknowledgement

The authors would like to thank Professor Peskin for his help, suggestions, and his lectures on simulations.

The source code and relevant documents of this project can be found here: github.com/Daniel-Chin/DiceSim

8. Appendix

8.1. Deriving Defferential Equations for Ball-Hill Experiment



Normal force must be perpendicular to the tangent:

$$F_{normal} = k \cdot (cos\theta, sin\theta)$$

where k is an unknown variable.

$$\therefore \Sigma F = (k \cdot cos\theta, k \cdot sin\theta - mg)$$

For the ball to stay on the surface, the centripetal acceleration must be provided by the projection of ΣF to the direction normal to the surface. In this way, we can solve $k = \left(g \cdot sin\theta - \frac{v^2}{R}\right)m$.

If we use cartesian coordinates in our phase space, we will have to solve k. However, if we use polar coordinates so that the phase space becomes $(\theta - \Omega)$, then k conveniently cancels out:

$$\begin{aligned} \frac{d\theta}{d^2t} &= \frac{\Sigma F \cdot (-sin\theta, cos\theta)}{mR} \\ &= \frac{-kcos\theta sin\theta + kcos\theta sin\theta - mgcos\theta}{mR} \\ &= -\frac{g \cdot cos\theta}{R} \\ &\propto -cos\theta \end{aligned}$$

where R is the radius of the hill.

9. References

Notes on Structural Dynamics with Stiff Links: The Backward-Euler Method, by Prof. C. Peskin.
www.math.nyu.edu/faculty/peskin/modsim_lecture_notes/backward_ euler.pdf

Notes on Interactions of Dynamic Structures with the Ground: Non–Penetration and Slicing Frictional Forces, by Prof. C. Peskin.
www.math.nyu.edu/faculty/peskin/modsim_lecture_notes/interaction_with_ground.pdf

Notes on Dynamics of Rigid Bodies, by Prof. C. Peskin.
www.math.nyu.edu/faculty/peskin/modsim_lecture_notes/rigid_body_motion.pdf

Notes on Structural Dynamics with Rigid Links, by Prof. C. Peskin.
www.math.nyu.edu/faculty/peskin/modsim_lecture_notes/mechanics_with_rigid_links.pdf