Simulación de Sistemas

Trabajo Práctico 2: Autómata Celular Off-Lattice

Bandadas de Agentes Autopropulsados

Baiges, Matías Sebastián - 59076
Comerci Wolcanyik, Nicolás - 59520
Piñeiro, Eugenia Sol - 59449

3 de septiembre de 2021

Índice general

0.1. Introducción

$$Ec_i = \frac{1}{2} m_i v_i^2 \quad Ep_{ij} = -G\frac{m_i m_j}{r_{ij}} \quad E_T = \sum_{i=1}^{n} Ec_i + \sum_{i=1}^{n}\sum_{j=i+1}^{n} Ep_i \quad \Delta E_T(\%) = \frac{E_{T_{final}} - E_{T_{incial}}}{E_{T_{inicial}}} \tag{1}$$

El objetivo del siguiente trabajo práctico es implementar un Autómata Celular *Off-Latice*, a fin de analizar el comportamiento de partículas puntuales en base a distintos parámetros.

El autómata *Off-Latice* se basa en un modelo matemático inspirado en el comportamiento colectivo de algunos Sistemas Naturales. Estos sujetos biológicos tienden a moverse con sus vecinos. Por ejemplo, bandadas, cardúmenes, manadas de cuadrúpedos o una colonia de bacterias.[2]

En cuanto al sistema, las partículas se desplazan con velocidad absoluta constante. Las mismas, en cada paso de tiempo, asumen el promedio de las direcciones de las partículas vecinas que se encuentran dentro de un radio r con un ruido añadido de forma aleatoria.

Este modelo exhibe un comportamiento cooperativo entre las partículas, las cuales a medida que va avanzando el tiempo comienzan a moverse con una simetría de forma espontánea.

La evolución temporal de las partículas viene dada por las siguientes ecuaciones:

$$x_i(t+1) = x_i(t) + v_i(t)\Delta t \tag{2}$$

donde x es la posición de la partícula, v la velocidad, y Δt el paso de tiempo.

$$\theta_i(t+1) = \langle \theta_i(t) \rangle_r + \Delta\theta \tag{3}$$

donde $\Delta\theta \epsilon [-\frac{\eta}{2}; \frac{\eta}{2}]$ es el ruido añadido, siendo η la amplitud del ruido, y $\langle \theta_i(t) \rangle_r$ es el promedio de los ángulos de todas las partículas en un radio r.

$$\langle \theta_i(t) \rangle_r = \frac{\langle sin(\theta_i(t)) \rangle_r}{\langle cos(\theta_i(t)) \rangle_r} \tag{4}$$

0.2. Implementación

La implementación del algoritmo *Off-Lattice* se desarrolló en el lenguaje de programación *Java*.

El flujo principal del programa (ver Figura 1) consiste en diversos pasos. En primer lugar, se obtienen los parámetros de configuración para iniciar la Simulación.

Luego, se generan partículas de forma aleatoria y se van actualizando tanto sus posiciones como sus ángulos con el algoritmo del autómata.

Por último, se guardan los resultados en distintos archivos que serán utilizados como *input* para generar la Animación en *Ovito* y realizar el postprocesamiento en *Python* para hacer un análisis de resultados.

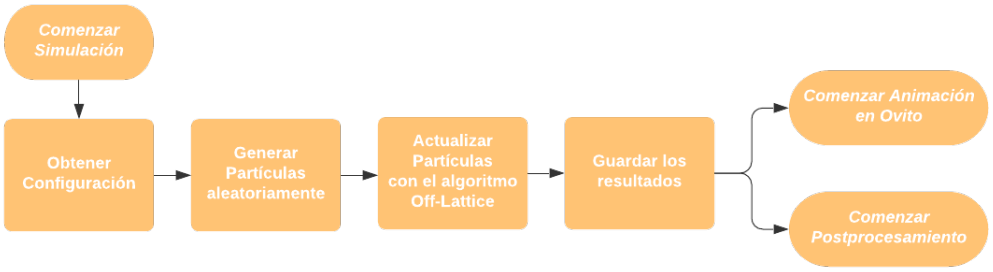


Figura 1: Diagrama de Flujo

En cuanto a la implementación de la Simulación, se cuenta con distintas clases en Java (ver Figura 2). La clase que implementa el flujo principal (main) es *ar.edu.itba.ss.OffLatticeSimulation*. Las partículas se modelaron con la clase *VelocityPaticle*, que extiende de una clase *ar.edu.itba.ss.Particle* y cuenta con su posición, ángulo y rapidez. Las mismas se generan a partir de la clase *VelocityParticleGenerator*.

Para calcular la polarización de las mismas se cuenta con la clase *ar.edu.itba.ss.Polarization*. El algoritmo del autómata se implementó en la clase *ar.edu.itba.ss.OffLattice* que hace uso de las mencionadas anteriormente.

A su vez, se cuenta con las clases para obtener los parámetros de configuración y obtención de resultados *ar.edu.itba.ss.ar.edu.itba.ss.ar.edu.itba.ss.ar.edu.itba.ss.Config*,

XYZWriter, ar.edu.itba.ss.JsonWriter y *ar.edu.itba.ss.Postprocessing*, sobre las cuales no se entrará en detalle ya que corresponden al postprocesamiento.

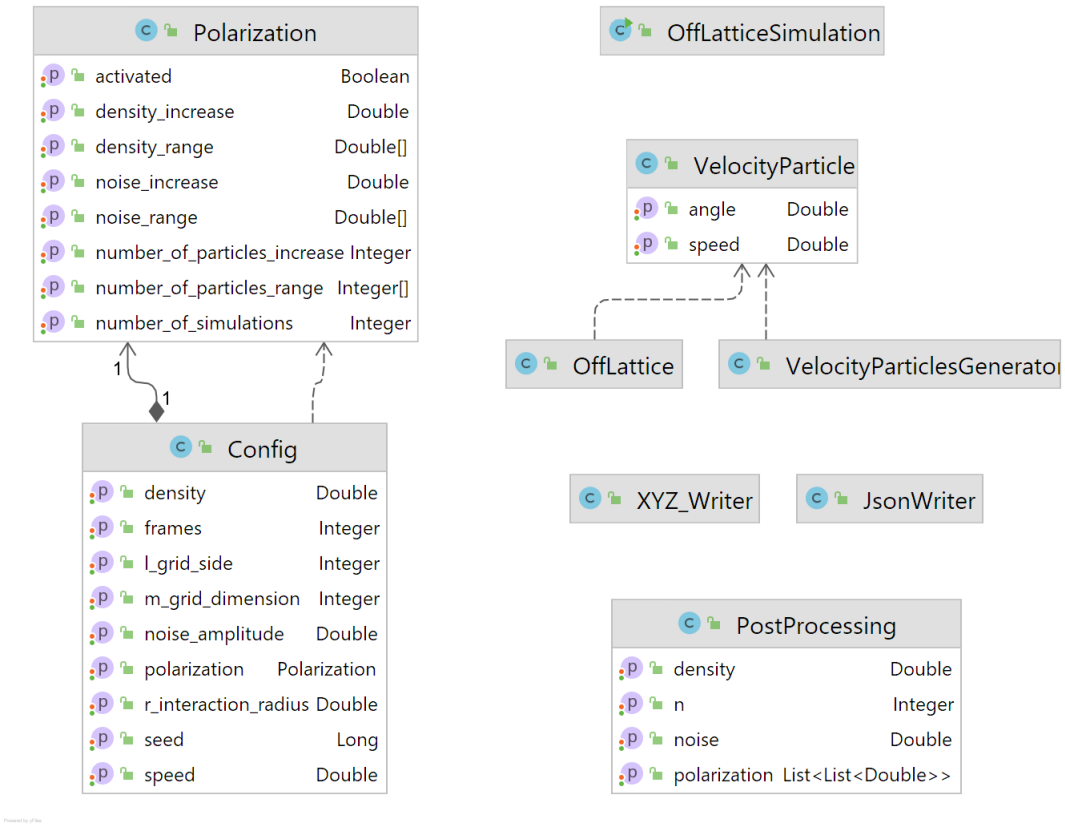


Figura 2: Diagrama UML

En el pseudocódigo correspondiente al flujo principal del programa (ver Algoritmo 1) se puede notar que se sigue el flujo mencionado previamente, en el cual se llama a la función *readValues()* para obtener la configuración del sistema, luego se generan partículas de forma aleatoria invocando la función *generateRandom()*, se llama al autómata Off-Lattice y por último se guardan los resultados en los archivos correspondientes.

En cuanto a la implementación del autómata (ver Algoritmo 2) cabe destacar que por cada pasa de tiempo (frame), en la línea 3 del pseudocódigo, se buscan los vecinos de las partículas a través del *Cell Index Method*. Este método define una grilla de dimensión MxM y se utilizan condiciones de contorno periódicas [1]. Luego, se actualizan el ángulo y la posición de cada partícula utilizando las ecuaciones de Evolución Temporal (1) (2) y (3).

Algorithm 1 Off-Lattice ar.edu.itba.ss.ar.edu.itba.ss.ar.edu.itba.ss.Simulation Main

```
1: procedure MAIN
2:     config ← mapper.readValues()
3:     particles ← ar.edu.itba.ss.ar.edu.itba.ss.ar.edu.itba.ss.VelocityParticlesGenerator.generate
4:     frames ← ar.edu.itba.ss.OffLattice.simulate(config)
5:     ar.edu.itba.ss.ar.edu.itba.ss.ar.edu.itba.ss.XYZ_Writer(FILENAME).addAllFrames(f
```

Algoritmo 1. Flujo Principal del Programa

Algorithm 2 Off-Lattice Algorithm

```
1: procedure SIMULATE
2: for frame in frames:
3:     neighbours ← ar.edu.itba.ss.CellIndexMethod.search()
4: for particle in neighbours + itself:
5:     newAngle ← atan2(sinAvg, cosAvg) + noise
6:     particle.setAngle(newAngle)
7:     particle.setX((x + speed * cos(newAngle)) mod L)
8:     particle.setY((y + speed * sin(newAngle)) mod L)
```

Algoritmo 2. Algoritmo del Autómata Off-Lattice

0.3. Simulaciones

El esquema del sistema (ver Figura 3) consta de una grilla cuyos lados son de longitud L, en la cual interactúan partículas con velocidad constante v y radio de interacción r.

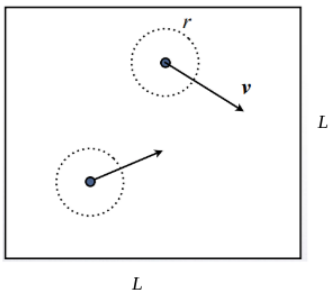


Figura 3: Esquema del Sistema

Para realizar las simulaciones se utilizaron los siguientes parámentros:

- Radio de interacción entre partículas $r = 1$
- Velocidad absoluta de las partículas $v = 0.03$
- Amplitud del ruido $\eta \in [0; 2\pi]$
- Lado de la grila L (Variable)
- Cantidad de partículas N (Variable)
- Densidad $\rho = \frac{N}{L^2}$
- Cantidad de pasos $frames = 2000$

En cada simulación se observó la polarización del sistema, la cual se define con la siguiente ecuación:

$$v_a = \frac{1}{Nv} \left| \sum_{i=1}^{N} v_i \right| \tag{5}$$

Si la polarización tiende a cero, las partículas se encuentran en total desorden, mientras que si la polarización tiende a uno se dice que las partículas están polarizadas.

Para correr las simulaciones se utilizarán las siguientes configuraciones:

1. Para analizar el ruido se utilizó N=300, ρ=0.6 y η variable
2. Para analizar la densidad se utilizó N=300, η=0.8 y ρ varibale

3. Densidad y ruido pequeños: N = 300, L = 25, ($\rho = 0.48$), $\eta = 0.1$
4. Densidad y ruido grandes: N = 300, L = 7, ($\rho = 6.12$), $\eta = 2.0$
5. Densidad grande y ruido pequeño: N = 300, L = 5, ($\rho = 12$), $\eta = 0.1$

0.4. Resultados

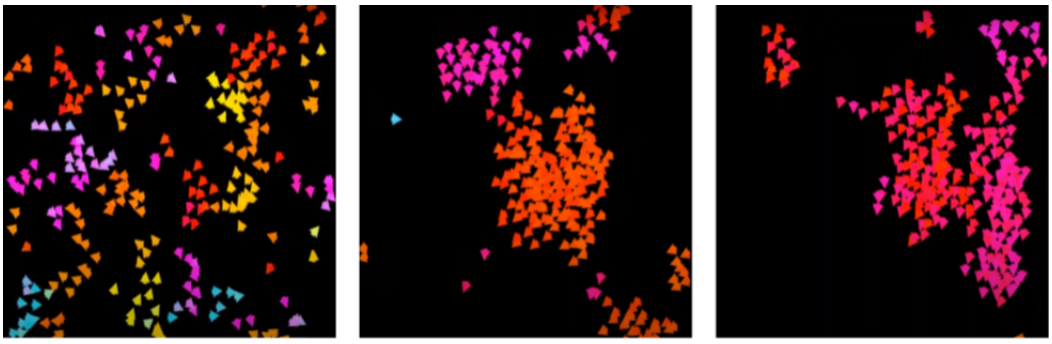
0.4.1. Análisis del Ruido

Luego de realizar simulaciones bajo las configuraciones propuestas (ver Figuras 4, 5 y 6), se observó que a medida que el ruido aumentaba, las particulas tendían a dispersarse, dirigiendose hacia distintas direcciones, es decir, manteniendo la cantidad de partículas y el ruido constante, *al aumentar el ruido, la polarización disminuye.*

En caso de poco ruido (ver Figura 4), se pudo observar como las partículas al cabo de un par de pasos (*frames*) se movían en la misma dirección (*partículas altamente polarizadas*).

Al aumentar el ruido (ver Figura 5), se observó que no se llegaba a una polarización tan clara como en el caso anterior, formándose grupos representados a través de los distintos colores.

Finalmente para $\eta = 5$ (ver Figura 6), se obtuvo una imágen invariable de múltiples colores y desorden a través de toda la simulación.



Frame inicial *Frame intermedio* *Frame final*

Figura 4: Capturas de los frames inicial, intermedio y final de una Animación con N=300; ρ=0.6 y η=0.5 Ver Animación

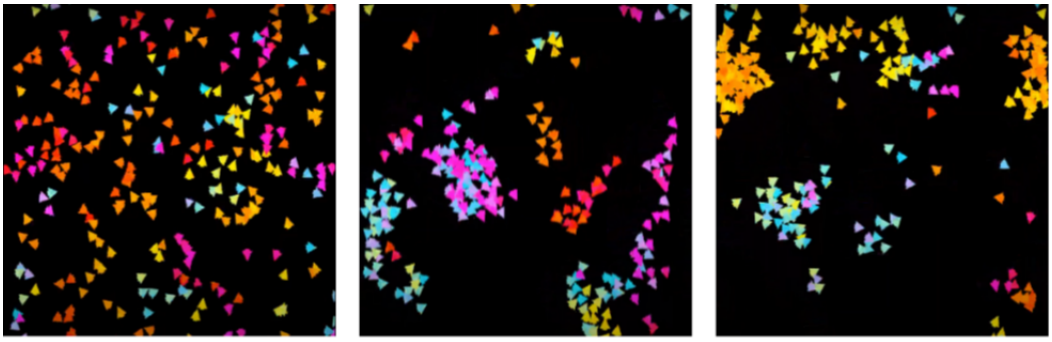


Figura 5: Capturas de los frames inicial, intermedio y final de una Animación con N=300; ρ=0.6 y η=1.5 Ver Animación

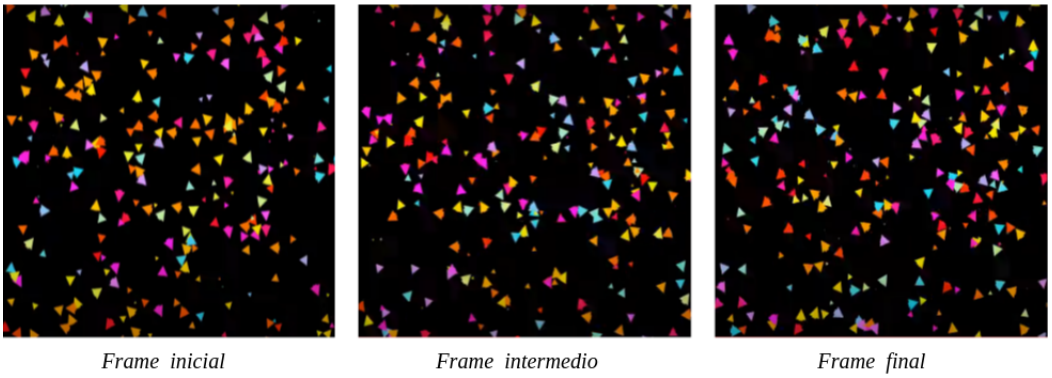


Figura 6: Capturas de los frames inicial, intermedio y final de una Animación con N=300; ρ=0.6 y η=5 Ver Animación

También se observó este mismo fenómeno al comparar la evolución de la polarización en función del tiempo para distintos valores de ruidos (ver Figura 7).

Además, se pudo ver esta situación para distintos valores de ruido (ver Figura 8), y para distintas cantidades de partículas (ver Figura 9), donde vimos que mientras menos partículas, mayor era la polarización manteniendo el ruido constante.

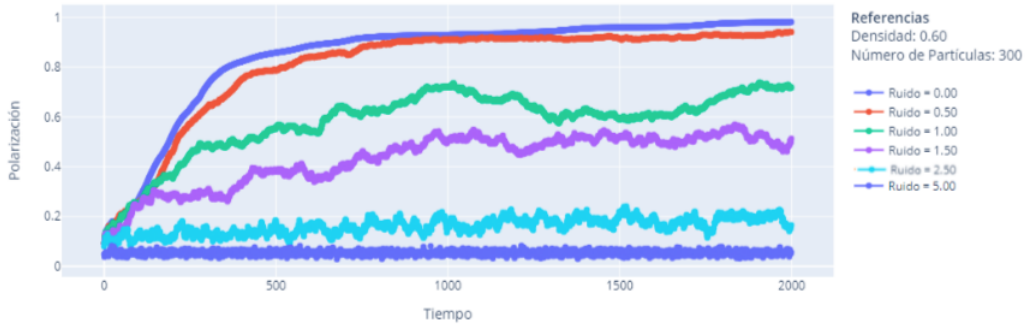


Figura 7: Polarización en función del tiempo para múltiples valores del ruido, ρ=0.6, N=300

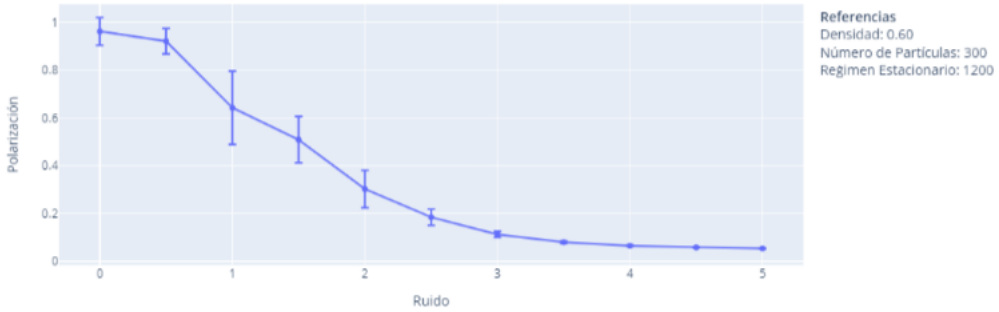


Figura 8: Polarización en función del ruido para ρ=0.6, N=300 y considerando el frame 1200 como comienzo del régimen estacionario.

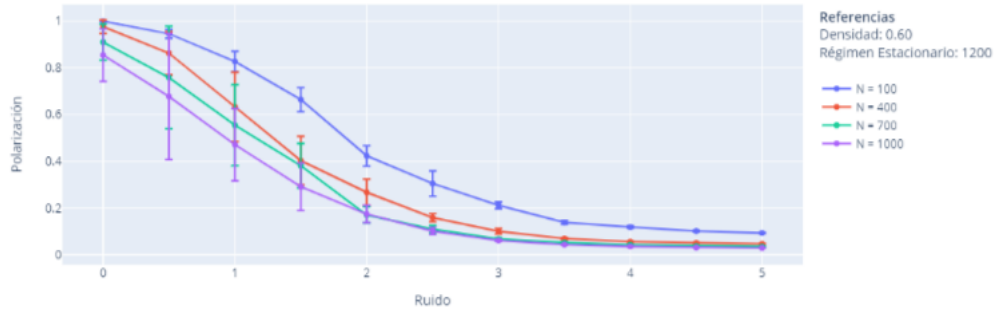


Figura 9: Polarización en función del ruido para múltiples valores de N con ρ=0.6 y considerando el frame 1200 como comienzo del régimen estacionario.

0.4.2. Análisis de la Densidad

Luego de realizar simulaciones bajo las configuraciones propuestas (ver Figuras 10, 11 y 12), se observó que a medida que se aumentaba la densidad de partículas, las partículas tienden a moverse de manera más ordenada, es decir, manteniendo la cantidad de partículas y el ruido constantes, *al aumentar la densidad de partículas, la polarización aumenta*.

En particular, para un densidad muy pequeña (ver Figura 10) se notó que las partículas comenzaban dispersas y luego de un tiempo se dividían en grupos. Dentro de cada uno de los grupos todas las partículas se desplazaban en la misma dirección.

En cambio, tomando un valor de densidad intermedio (ver Figura 11) se observó que las partículas al principio generaban pequeños grupos pero luego comenzaban a desplazarse todas en las mismas dirección.

Por último, tomando un valor de densidad grande (ver Figura 12) se pudo notar que rápidamente las partículas se acomodaban en la misma dirección.

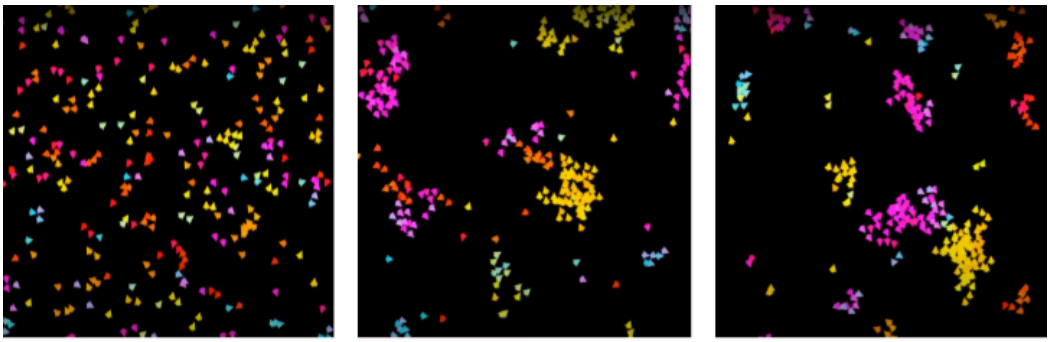


Figura 10: Capturas de los frames inicial, intermedio y final de una Animación con N=300; ρ=0.25 y η=0.8 Ver Animación

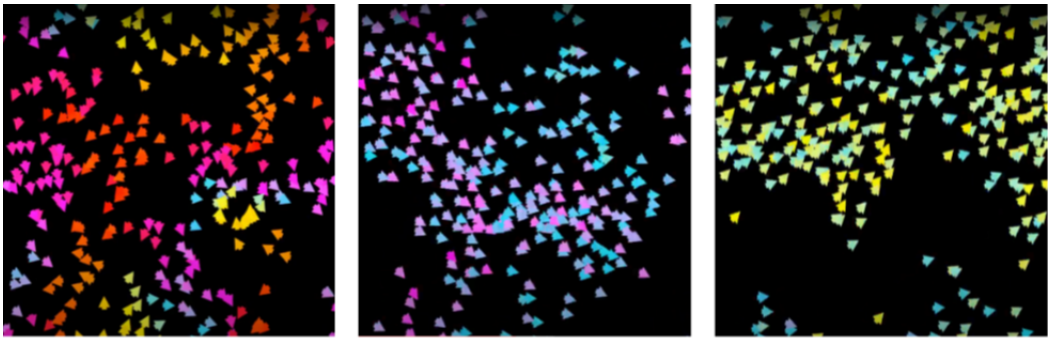


Figura 11: Capturas de los frames inicial, intermedio y final de una Animación con N=300; ρ=3 y η=0.8 Ver Animación

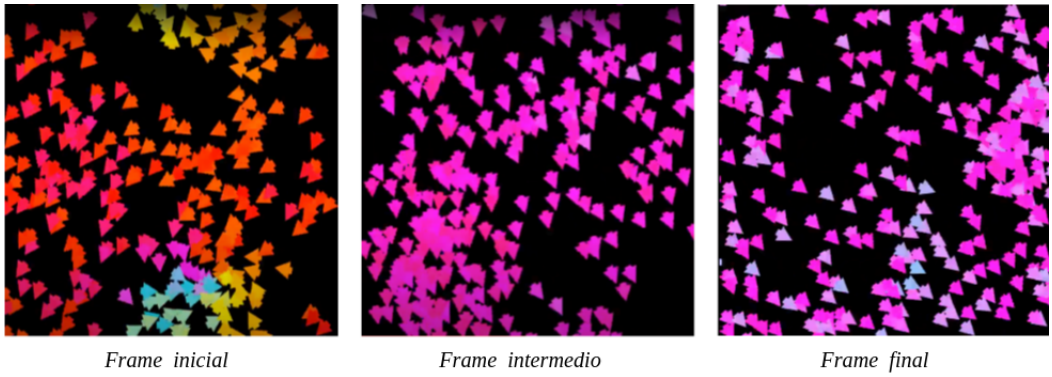


Figura 12: Capturas de los frames inicial, intermedio y final de una Animación con N=300; ρ=6 y η=0.8 Ver Animación

Analizando la polarización en función del tiempo para múltiples valores de densidad, dejando el resto de los parámetros constantes (ver Figura 13), se puede apreciar que a medida que se aumentan los valores de densidad, aumenta la polarización a lo largo del tiempo. Esto se corresponde con lo mencionado previamente sobre las partículas alineandose (aumenta la polarización) a medida que aumenta la densidad.

Cabe destacar que mientras más bajo es el valor de la densidad, se notó que se necesitaban más frames para alcanzar un régimen estacionario. En este caso se decidió promediar los valores de polarización a partir del frame 1200 para graficar la polarización en función de la densidad para un ruido determinado (ver Figura 14) debido a que aproximadamente a partir de allí los valores de polarización son similares.

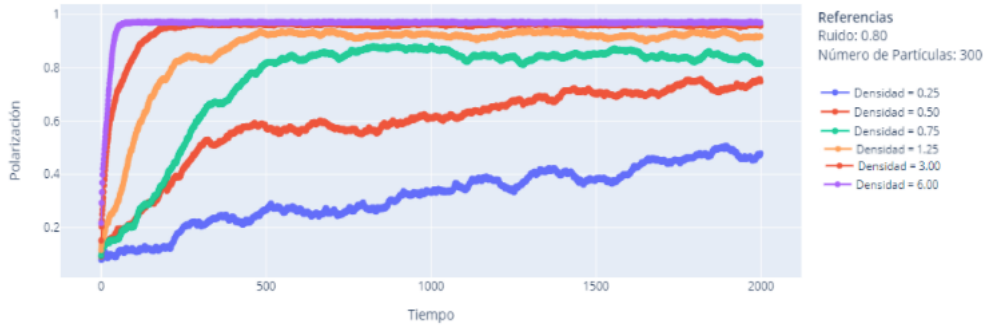


Figura 13: Polarización en función del tiempo para múltiples valores de densidad con N=300; η=0.8

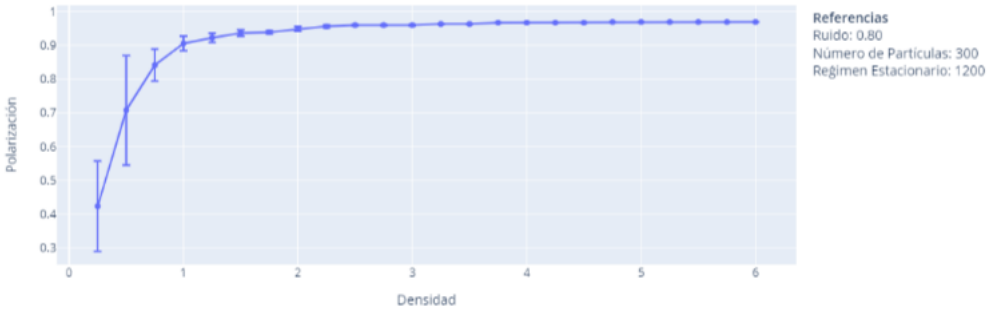


Figura 14: Polarización en función de la densidad con N=300; η=0.8 usando el frame 1200 como comienzo del régimen estacionario.

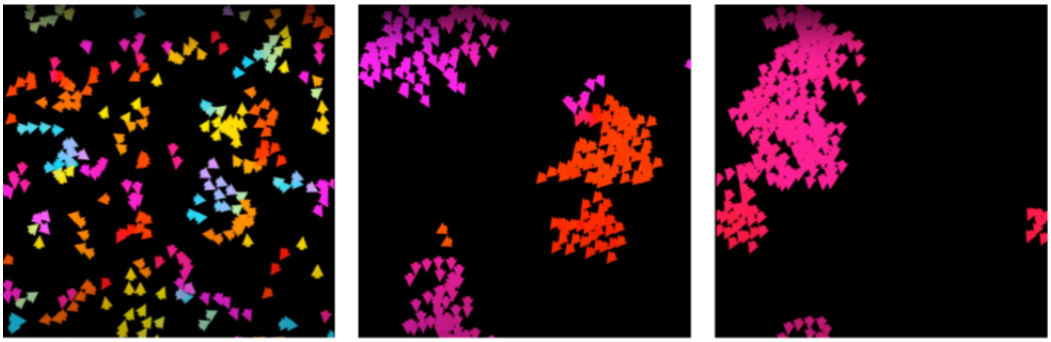
0.4.3. Comportamientos Destacados

Se utilizaron distintas configuraciones propuestas en el paper [2] con el objetivo de evaluar distintos comportamientos destacados del sistema.

En primer lugar, se utilizaron valores de densidad y ruido pequeños y se pudo notar que las partículas tienden a formar grupos moviendose de forma coherente en direcciones aleatorias (ver Figura 15)

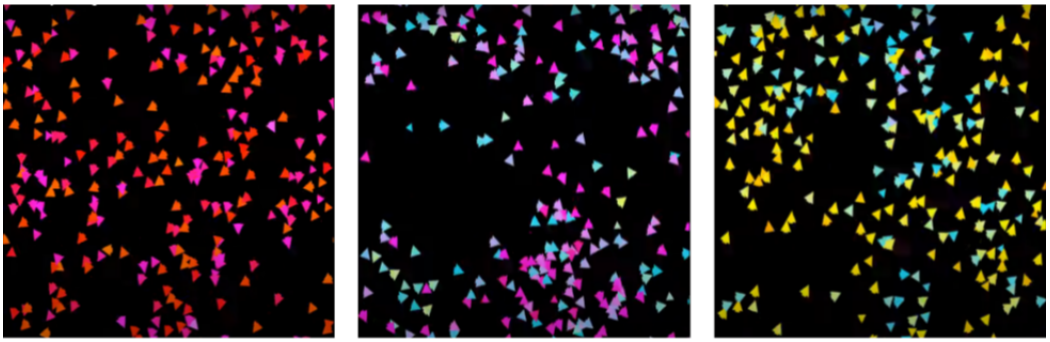
Luego, para valores de densidad y ruido grandes (ver Figura 16) se notó que las partículas se desplazan de forma aleatoria pero con cierta correlación.

Por último, con una densidad grande y un ruido pequeño (ver Figura 17) se notó que el movimiento se vuelve ordenado para todas las partículas.



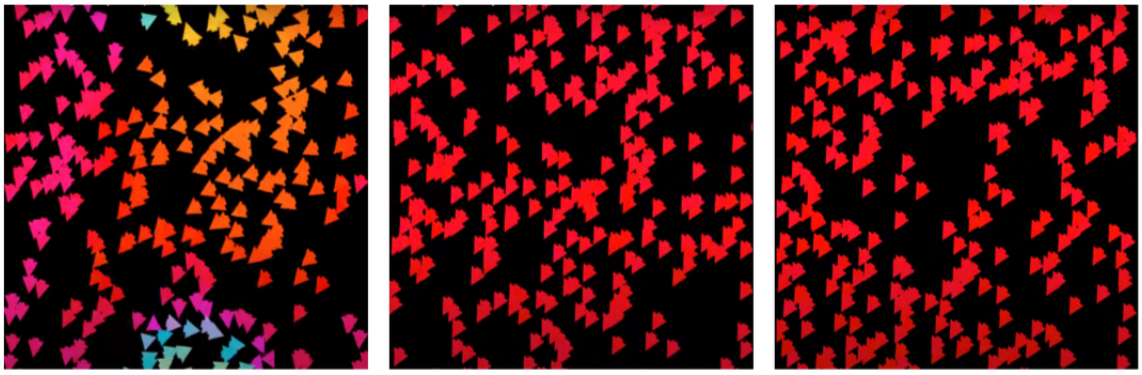
Frame inicial *Frame intermedio* *Frame final*

Figura 15: Capturas de los frames inicial, intermedio y final de una Animación con N = 300; L = 25; ($\rho = 0.48$); $\eta = 0.1$ Ver Animación



Frame inicial *Frame intermedio* *Frame final*

Figura 16: Capturas de los frames inicial, intermedio y final de una Animación con N = 300; L = 7; ($\rho = 6.12$); $\eta = 2.0$ Ver Animación



Frame inicial *Frame intermedio* *Frame final*

Figura 17: Capturas de los frames inicial, intermedio y final de una Animación con N = 300; L = 5; ($\rho = 12$); $\eta = 0.1$ Ver Animación

0.5. Conclusiones

Teniendo en cuenta los resultados obtenidos a partir de las simulaciones, se llegó a las siguientes conclusiones:

- Manteniendo la cantidad de partículas y la densidad constantes, al aumentar el ruido disminuye la polarización.
- Manteniendo la cantidad de partículas y el ruido constantes, al aumentar la densidad de partículas aumenta la polarización.
- El sistema presenta distintos comportamientos según los parámetros elegidos.

Bibliografía

[1] Xiang-Yun Guo and Pascal Brault. Early stages of silicon nitride film growth studied by molecular dynamics simulations. *Surface science*, 488(1-2):133–140, 2001.

[2] Tamás Vicsek, András Czirók, Eshel Ben-Jacob, Inon Cohen, and Ofer Shochet. Novel type of phase transition in a system of self-driven particles. *Physical review letters*, 75(6):1226, 1995.