



Simulación de Multitudes

Grupo 7
Duffau, Teófilo Manuel
Lynch, Ezequiel



Fundamentos



Fundamentos


Dinámica peatonal en condiciones competitivas:

- El sistema contiene partículas egoístas que tratan de salir lo antes posible de un recinto con una sola salida.
- Las interacciones entre partículas son de repulsión y de intento de esquivar.



Modelo Matemático

Contractile Particle Model



En este modelo las partículas tienen un radio variable en el tiempo y de este radio depende la velocidad.

Cuando contacta con otra partícula, sus radios colapsan al mínimo y su velocidad cambia a la dirección opuesta a la otra partícula y al módulo de la velocidad máxima.

Cada partícula tiene un destino al cual apunta su vector velocidad.

Cálculo de posición, velocidad y radio

Posición: $\overline{x}(t + dt) = \overline{x}(t) + \overline{v_d} * dt$

Velocidad: $|\overline{v_d}| = v_d^{max} [(r - r_{min}) / (r_{max} - r_{min})]^\beta$

Radio: $r(t + d_t) = \min(r(t) + r_{max} / (\tau / \Delta t), r_{max})$

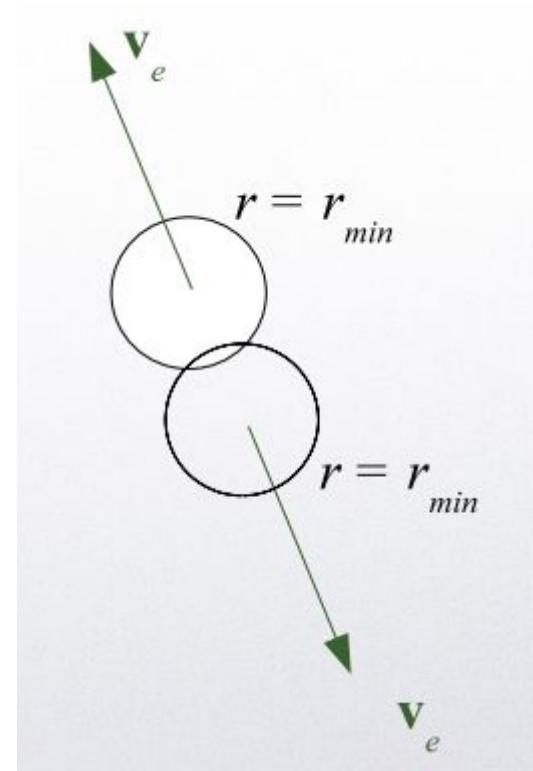
Contacto

Cuando 2 partículas entran en contacto aparece una velocidad de escape en sentido contrario al centro de la partícula con la que hace contacto y los radios se contraen al mínimo

$$|\overline{v_d}| = v_{max}$$

$$\overline{v_d} = \overline{v_e}$$

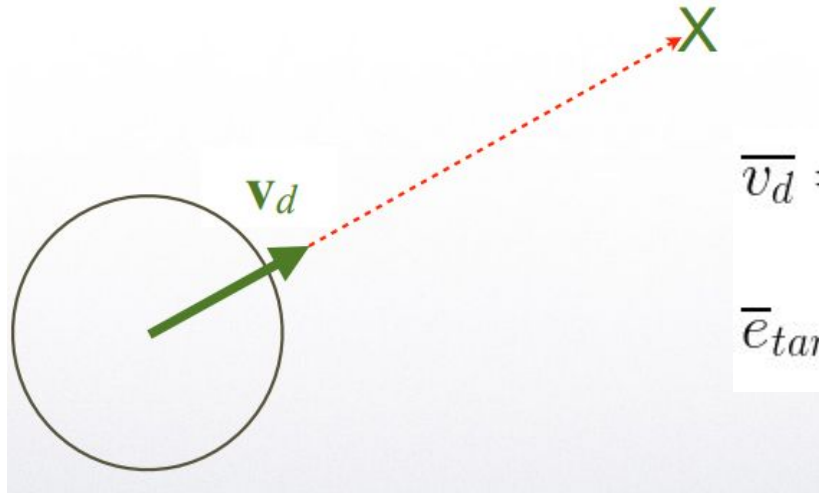
$$r = r_{min}$$



Cálculo de dirección de velocidad y objetivo

Las partículas tienen como target la posición (19.5, 10.0) hasta que se sitúan sobre la puerta que tienen como target salir (30.0, y).

```
void calculateTarget(){  
    if((x > 19 && y > 9.6 && y < 10.4) || x > 20){  
        target = new Target( x: 30, y);  
    } else {  
        target = new Target( x: 19.5, y: 10);  
    }  
}
```



$$\overline{v_d} = v_d * \overline{e}_{target}$$

$$\overline{e}_{target} = (\overline{x}_{target} - \overline{x}) / |\overline{x}_{target} - \overline{x}|$$



Cálculo del caudal

Para el cálculo del caudal se guardó cada tiempo de salida de las partículas que atravesaban la ranura. Con estos datos se aplicó una ventana móvil de 5 segundos a la cual luego se le calculó la media móvil con otra ventana de 20 elementos. La ecuación utilizada para calcular el caudal fue

$$Q[i] = \frac{(T[i + N] - T[i])}{N}$$

siendo $T[]$: un vector que contiene el flujo calculado en cada instante basado en los 5 segundos pasados.



Implementación



Modelo

- **MultitudeParticle:** representa la partícula y tiene toda la información que la concierne.
- **MultitudeGrid:** es la encargada de mantener el cell index method y calcular los vecinos.
- **SimulatorMultitude:** es la simulación en sí. Con métodos para generar las partículas, calcular el siguiente paso de la simulación e imprimir a archivo cuando ella terminase.



Algoritmo

1. Se generan las partículas con distribución uniforme y se agregan al MultitudeGrid.
2. En cada paso:
 - a. Se calculan los vecinos usando cell index method y se actualizan las velocidades y los radios si hubiera contacto
 - b. Se chequea si la partícula está en contacto con una pared.
 - c. Se recalcula el target.
 - d. Se actualizan las posiciones, las velocidades y los radios
 - e. Se cuenta cuántas partículas quedan en el recinto
3. Se corta el ciclo al quedar 0 partículas dentro.
4. Luego se guarda el archivo.

Pseudocódigo

Main():

 Simulator.simulate(deltaTime)

Simulator.simulate(deltaTime):

 deltaTime2 = 1 / (deltaTime / 60) //para que en tiempo real sean 60fps

 generateParticles() //genera hasta que no puede generar sin overlap en 100000 intentos

 counter = 0

 while(true):

 checkWall() // chequea para cada partícula si está en contacto con alguna pared

 calculateTarget() // recalcula el target de las partículas

 updateParticles() // calcula las nuevas posiciones de las partículas y sus radios y velocidades

 recalculateNeighbours() // utilizando cell index method y si hay contacto setea velocidades y radios

 if(counter % deltaTime2):

 saveState()

 if(countInsideParticles() == 0): // cuenta partículas y si salieron todas termina

 printToFile()

 return



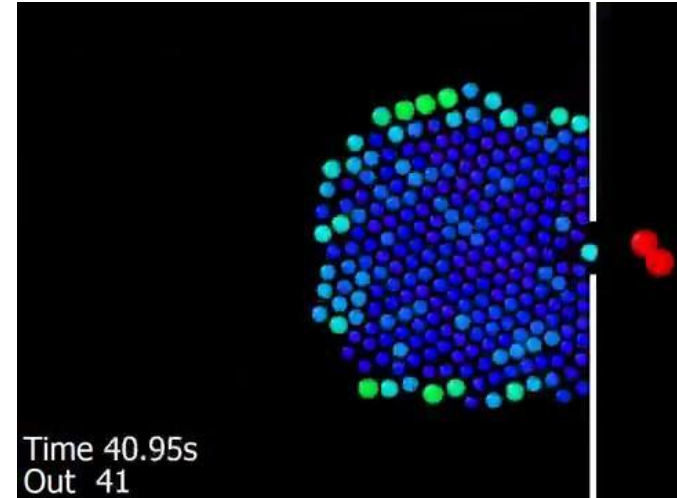
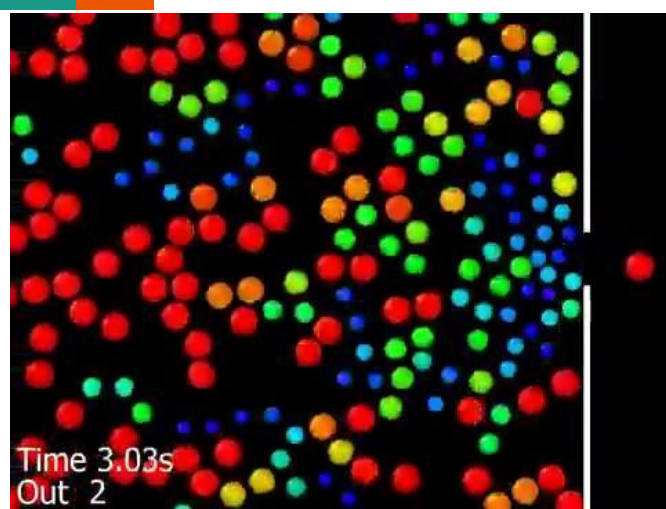
Resultados

Parámetros y valores iniciales



- Para llevar a cabo las simulaciones se utilizó un tiempo delta de $1e-2$ y para cada prueba se hicieron 5 simulaciones con $\tau = 0.4s$, $\beta = 1$, $r_{\min} = 0.15m$, $r_{\max} = 0.32m$, v_{\max} variable
- Las simulaciones en su totalidad fueron corridas en un recinto de 20m de alto X 20m de ancho y con un tamaño de ranura de 1.2m y cantidad de partículas igual a 300.
- Como Δt se utilizó $1e-2s$.

Animaciones



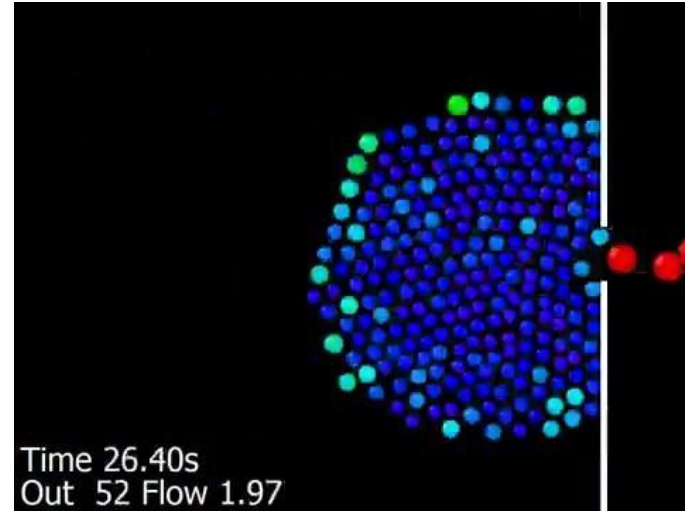
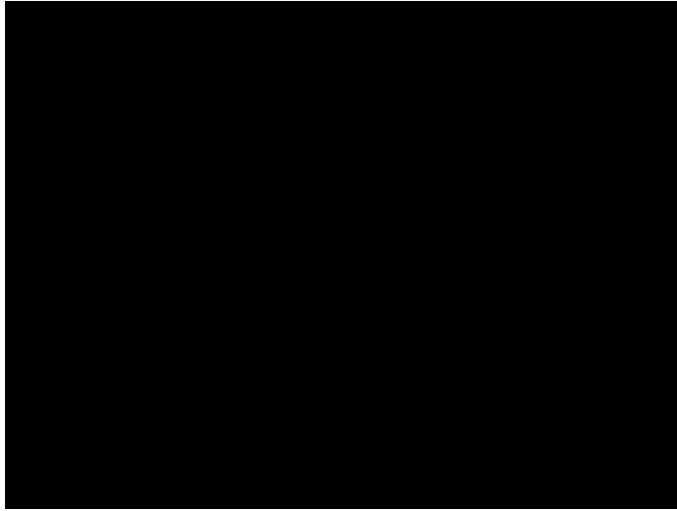
$$v_{\max} = 1.55 \text{ m/s}$$

$$r_{\min} = 0.15 \text{ m}$$



$$r_{\max} = 0.35 \text{ m}$$

Animaciones



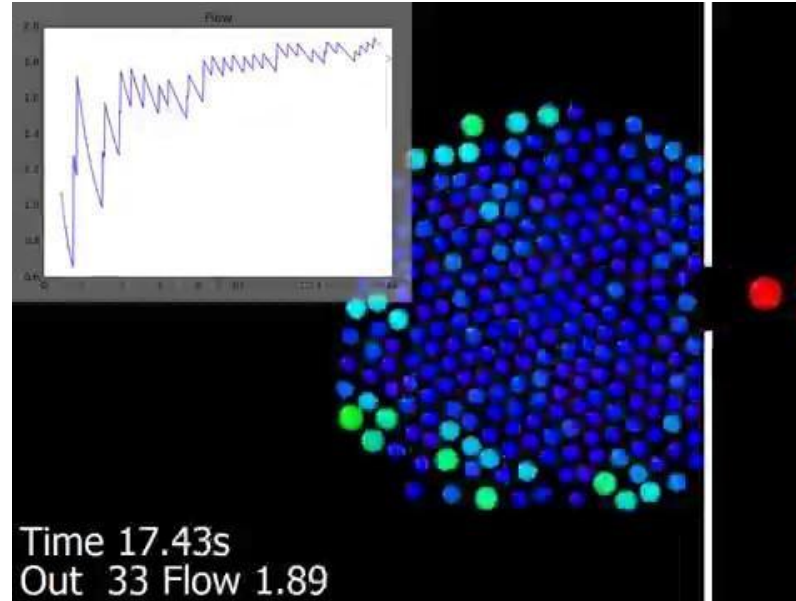
$$v_{\max} = 3\text{m/s}$$

$$r_{\min} = 0.15\text{m}$$



$$r_{\max} = 0.35\text{m}$$

Animaciones



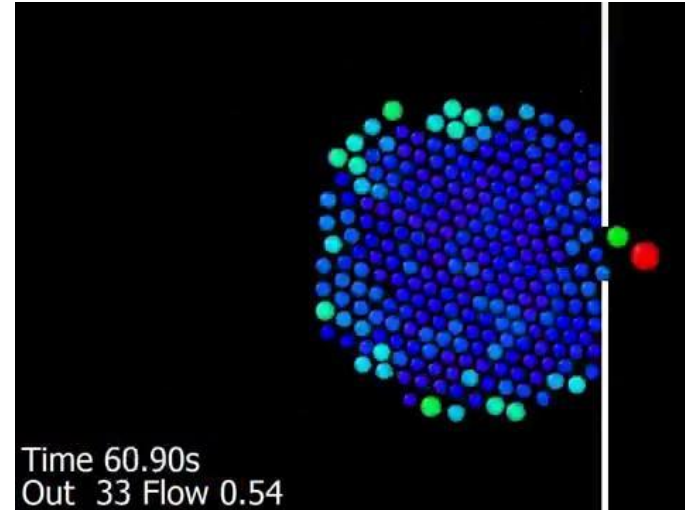
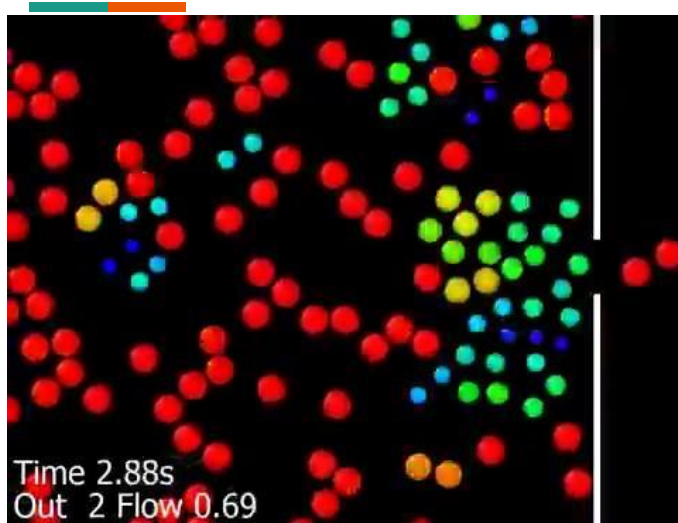
$$v_{\max} = 3\text{m/s}$$

$$r_{\min} = 0.15\text{m}$$



$$r_{\max} = 0.35\text{m}$$

Animaciones



$$v_{\max} = 0.75\text{m/s}$$

$$r_{\min} = 0.15\text{m}$$

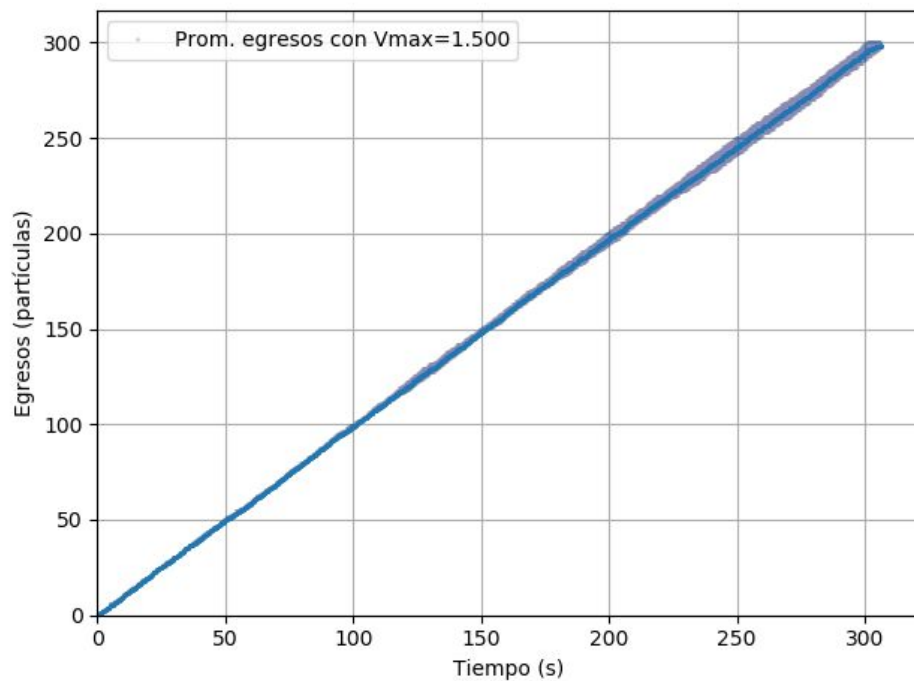
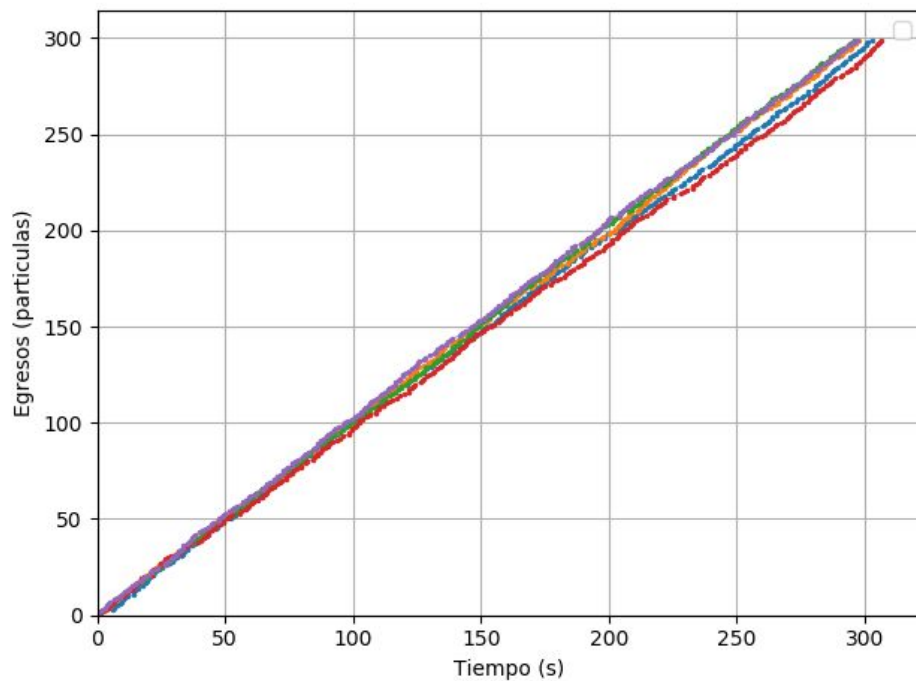


$$r_{\max} = 0.35\text{m}$$

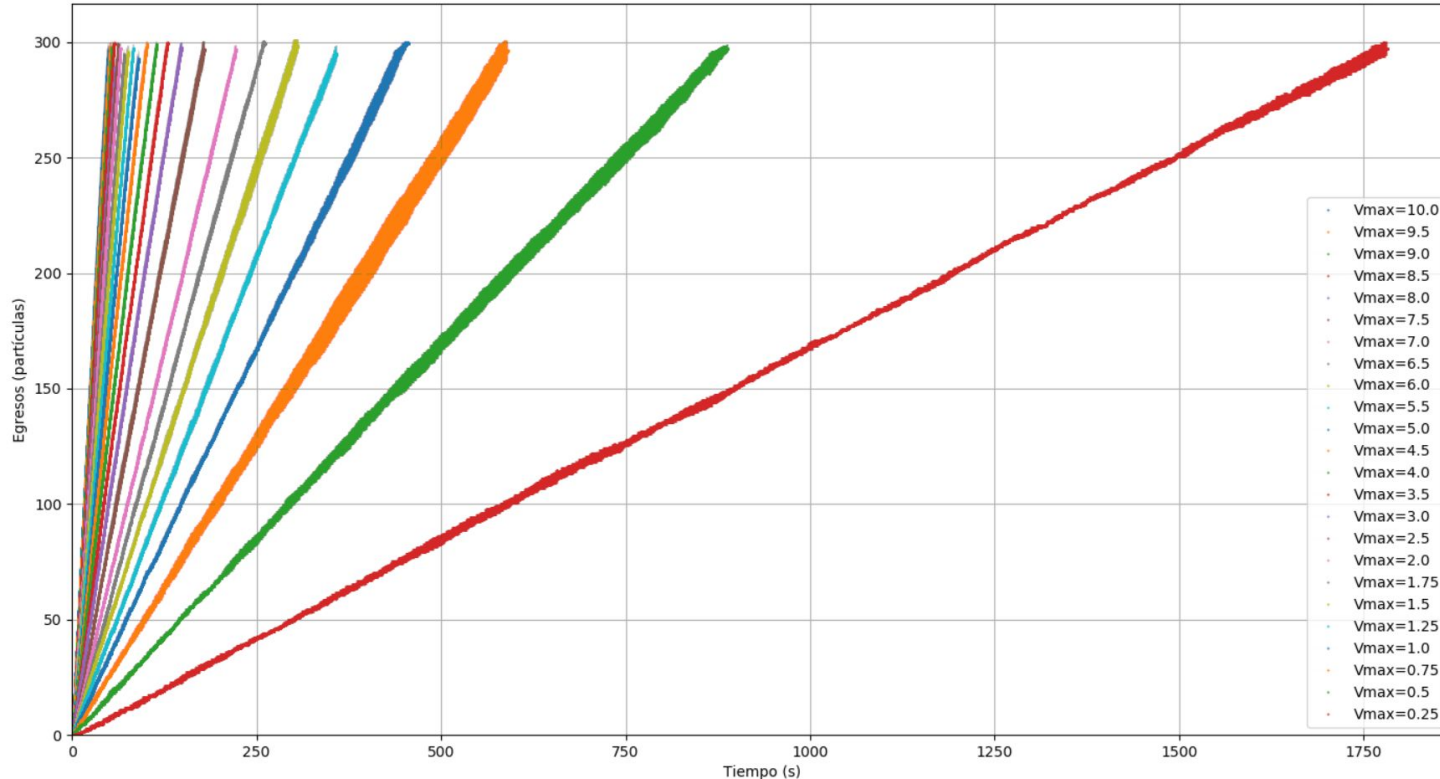
Egresos



5 corridas con $v_{\max} = 1.5$ m/s

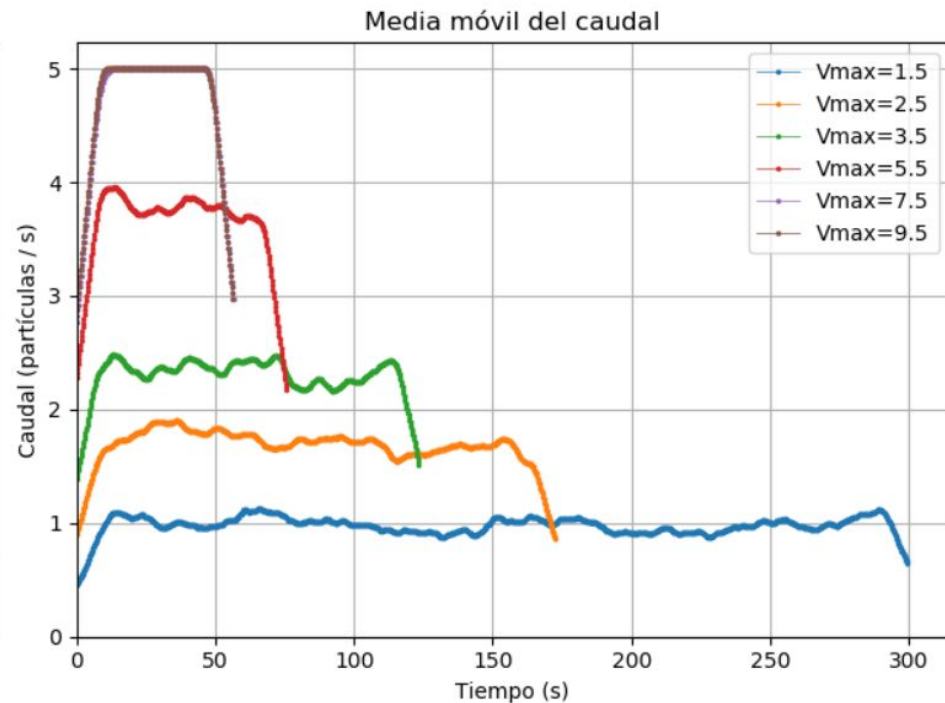
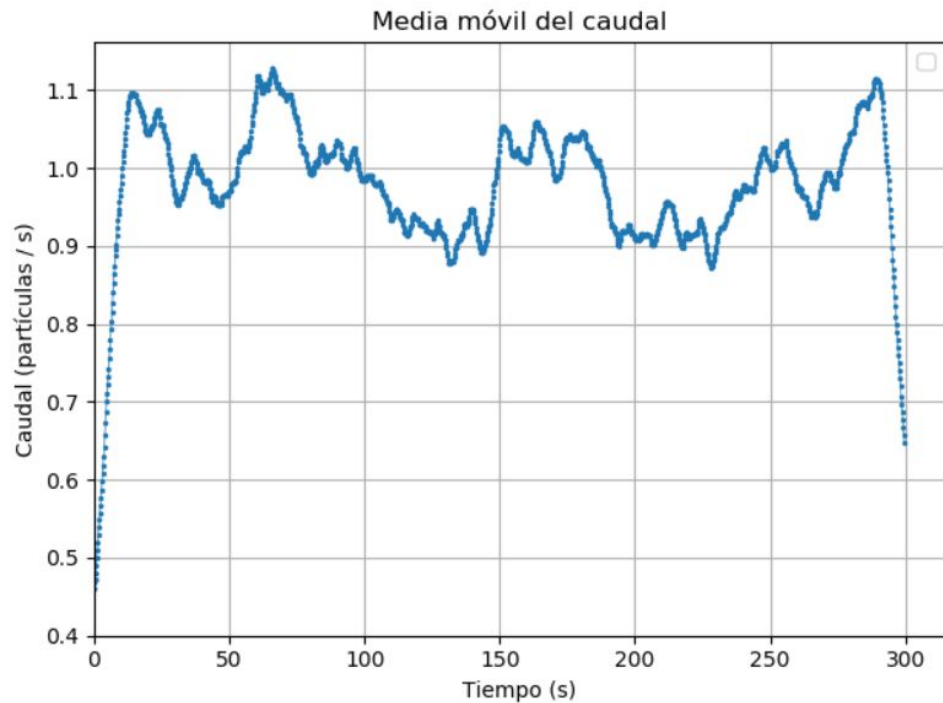


Promedio de egresos dependiendo de v_{\max}

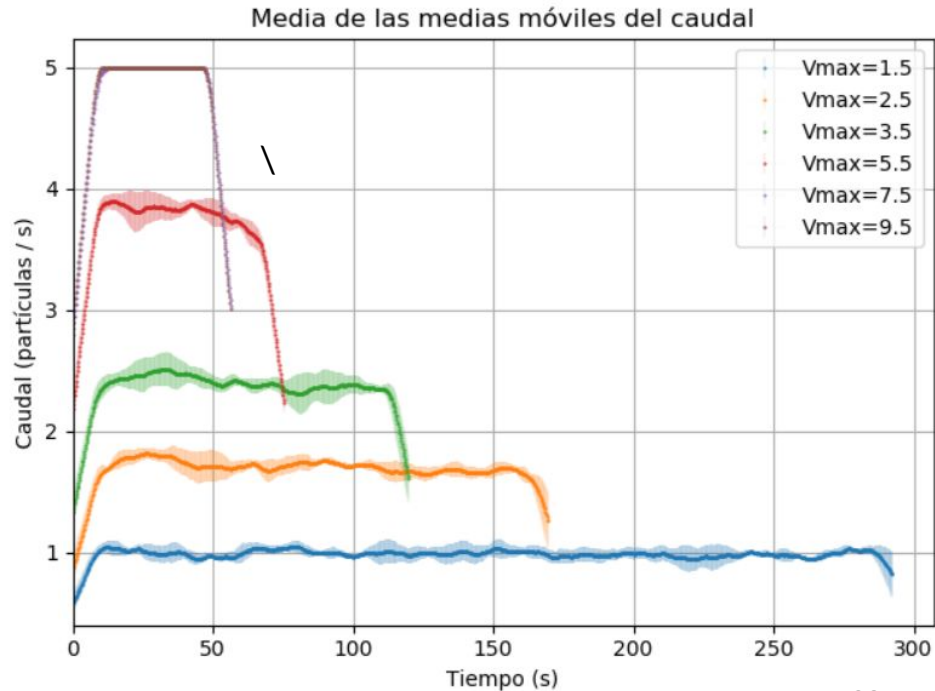
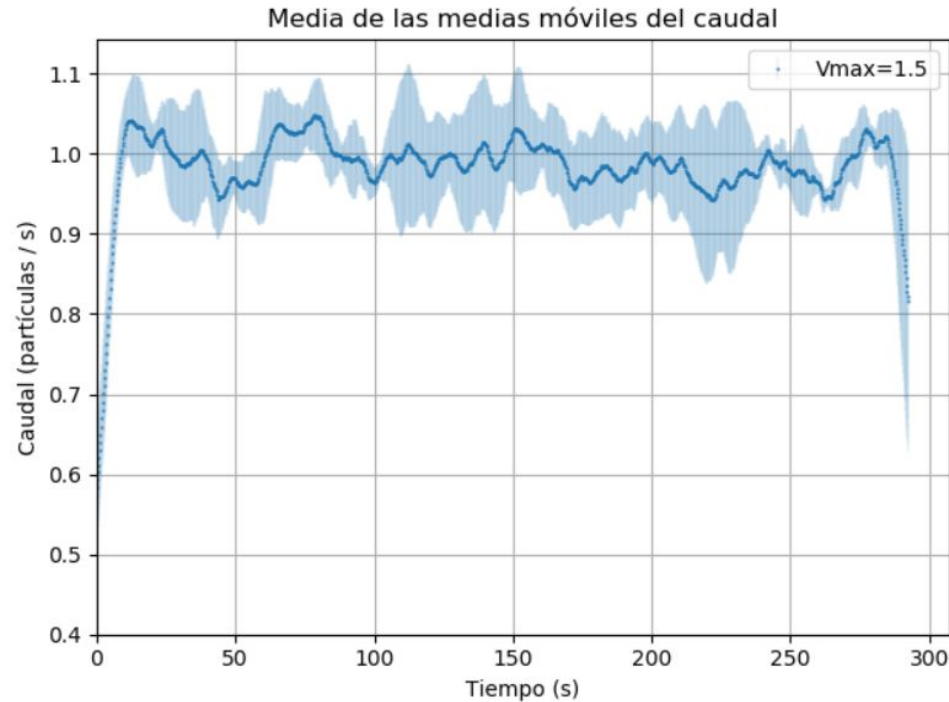


En este gráfico se presenta la media y su error de las corridas del gráfico anterior.

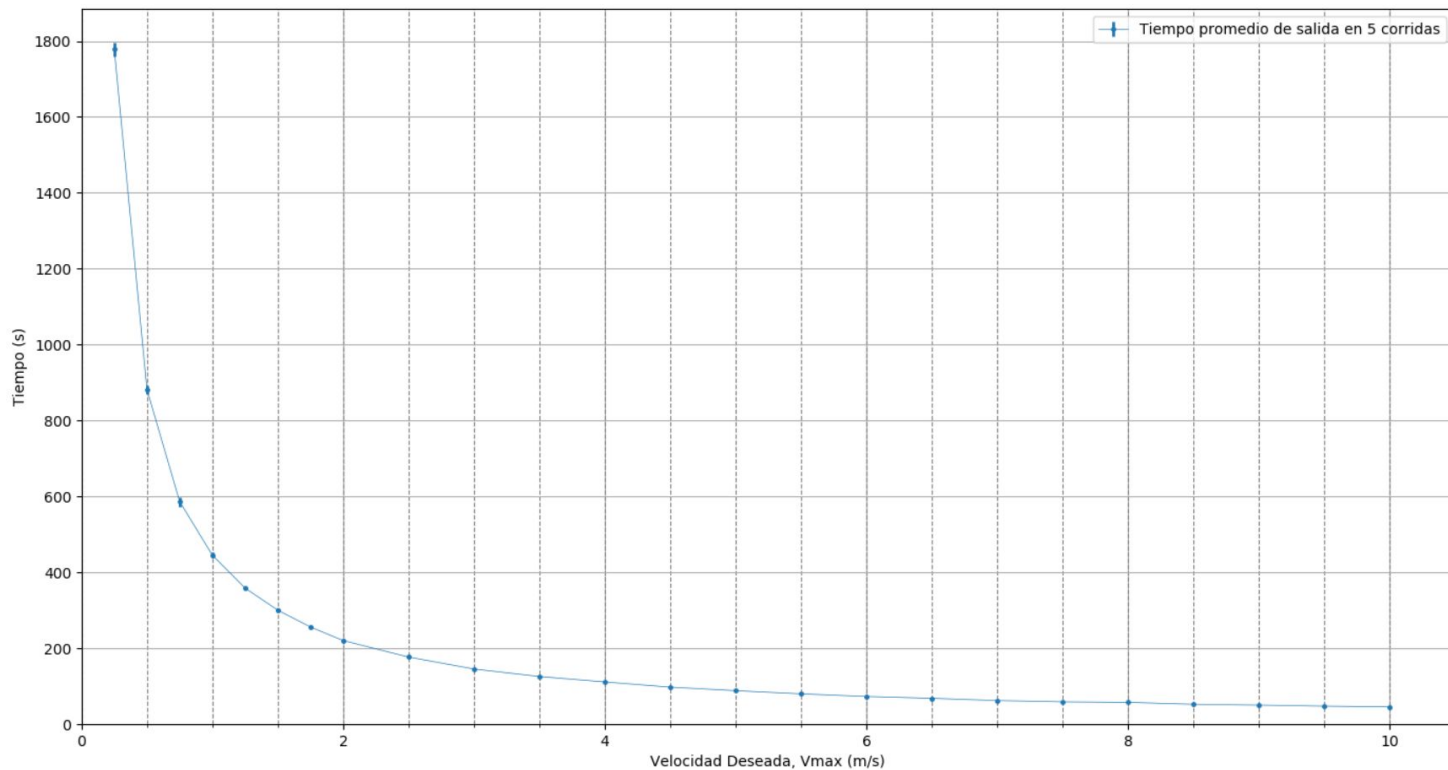
Caudal para $v_{\max} = 1.5\text{m/s}$ y otras



Caudal promedio para $v_{\max} = 1.5\text{m/s}$ y otras



Relación entre tiempo y velocidad deseada





Conclusiones



Conclusiones

- A medida que aumenta la velocidad máxima de las partículas, el tiempo promedio de egreso disminuye.
- El flujo de salida de las partículas es estable a lo largo del tiempo.
-