Investigating the Aerodynamic Impact of Slipstreams on Car Drag Through the Lattice Boltzmann Method

Luke Johnson
Level 3 MSci. Laboratory, School of Physics, University of Bristol.
(Dated: November 26, 2023)

This experiment aimed to utilise the Lattice Boltzmann Method in conjunction with computational techniques to simulate a fluid interacting with a moving car and reproduce its slipstream. This was achieved by using Python to recreate a mesoscopic model of the system in discrete lattice space to measure the drag on a following car at various separations. It was found that for the Cybertruck model of car, following at a distance of 123 cm gave a 48.3% reduction in drag while maintaining a good balance between performance and safety in the context of racing.

INTRODUCTION

The main purpose of this experiment was to successfully recreate the behaviour of fluid flowing past an obstacle using computers. The investigation aimed to solve various Lattice Boltzmann equations and apply this theory to real world motorsport racing. In many physical and racing sports, speed is the defining factor for victory and in many cases the main way of increasing speed and efficiency is by minimising drag. Drag force, F_D, is proportional to the square of the speed at which the object is moving through the fluid [1],

$$F_D = \frac{\rho v^2 C_D A}{2}, \tag{1}$$

where ρ is the density of the fluid, v is the speed of the object relative to the fluid, C_D is the drag coefficient and A is the cross-sectional area.

The idea of the slipstream was first used in cycling and can now be seen in the team pursuit Olympic event held in the velodrome. According to the Journal of Sports Sciences [2], it is estimated that 80% of total resistance encountered by the cyclists is from air resistance and the athletes saved 25% more energy when utilising slipstreams compared to solo events where they cannot. Furthermore, a research project which examined the aerodynamic drag in the middle to rear region of cycling pelotons (the main pack of riders in a road cycling race) found that the drag reduced down to 5-10% compared to an isolated rider [3].

In Formula 1, slipstreaming leads to greater car speeds, faster lap times and overtaking opportunities, so the motivation to study it for the purpose of maximising competitive advantage is high. To achieve the greatest slipstreaming effect, a driver needs to position their car in the turbulent air just behind the rear wing of the car in front as this is where the low pressure zone is strongest. This allows the driver to 'slingshot' past the car in front. One of the best implementations of the strategy can be seen in the 2000 Belgian Grand Prix [4] when Mika Hakkinen famously overtook two cars at once with the extra power saved by slipstreaming which consequently allowed him to win the race.

The optimal following distance for a car will be explored using a simulated wind tunnel in Python and the Lattice Boltzmann method which will then be compared to real life data.

It is important to study the benefit of slipstreaming as it is not without risks. If the cars are travelling too close at high speeds, there is a danger of collision which can cause a serious crash. This risk is exacerbated by the extra instability caused when driving in the turbulent wake of another vehicle, making the car harder to control.

THEORY

Slipstreaming is only possible when the airflow behind the cars becomes turbulent instead of laminar. Laminar flow is characterised by smooth, uniform fluid flow whereas turbulent flow occurs when the motion becomes chaotic and random due to the high inertial forces of the fluid. Vortices are only produced in turbulent flow which happens when the Reynolds number, Re, is much larger than 1. This dimensionless quantity is given as

$$Re = \frac{\rho u_0 L}{\mu}, \tag{2}$$

where ρ is the fluid density, u_0 is the flow speed, L is the characteristic length scale and μ is the dynamic viscosity. Airflow becomes turbulent at Re values greater than 3500 [5] which is easily exceeded by F1 cars that can produce Re up to 2.2×10^5 according to a study [6] that used advanced 3D software to test the aerodynamics of an F1 car with immense accuracy.

The Lattice Boltzmann method (LBM) is a mesoscopic approach which discretises space into a lattice structure [7] with a finite resolution and calculates the probability distribution, *f*, of particles over phase-space as a function of time,

$$\frac{d}{dt} f(\mathbf{r}, \mathbf{v}, t) = \Omega(f). \tag{3}$$

The derivative term describes the physical motion and external forces on the fluid and Ω is the source term describing collisions.

By finding the discrete moments of distributions, calculations on the microscopic scale can be extended to the macroscopic system. This can be done by discretising space and time into steps:

$$f_j(\mathbf{x}_i + \mathbf{c}_j \Delta t, t + \Delta t) = f_j(\mathbf{x}_i, t) + \Omega_j(f). \tag{4}$$

Here, after each time step $t + \Delta t$ the particles will move a distance of one site to a neighbouring site $\mathbf{x}_i + \mathbf{c}_j \Delta t$ in the 2D lattice. $\mathbf{x}_i$ is the position in the lattice of a particle and $\mathbf{c}_j$ has 9 values for velocity (one pointing to each of the surrounding lattice points).

The role of the Ω term at each site is to conserve mass and momentum, described by the Bhatnagar Gross Krook (BGK) operator,

$$\Omega_j(f) = -\frac{f_j(\mathbf{x}_i, t) - f_j^{eq}(\mathbf{x}_i, t)}{\tau}\Delta t, \tag{5}$$

where τ is a positive value that describes the rate at which the particle distributions relax back to the equilibrium distribution f_j^{eq}. It is now possible to substitute Eq. 5 into Eq. 4 and split it into two parts: the collision step and the streaming step shown respectively below:

$$f_j^*(\mathbf{x}_i, t) = (1 - \frac{\Delta t}{\tau})f_j(\mathbf{x}_i, t) + \frac{\Delta t}{\tau}f_j^{eq}(\mathbf{x}_i, t) \tag{6}$$

$$f_j(\mathbf{x}_i + \mathbf{c}_j\Delta t, t + \Delta t) = f_j^*(\mathbf{x}_i, t). \tag{7}$$

The distribution after the collision is represented by f_j^* (calculated in Eq. 6) and is responsible for updating the distribution function based on local collisions between the particles. The streaming step from Eq. 7 essentially passes information about the velocity distribution from the current, central site to its neighbouring ones which in turn affects and updates the values at their lattice sites. It also considers how the neighbouring sites' distributions affect the central site we are considering.

The LBM algorithm contains information for the amount of fluid at each lattice point, stored in the f array, as well as the average velocity at that location in the lattice, stored in the **u** array. With this known data, it is therefore possible to calculate the force at each lattice point acting on an obstacle by applying Newton's 2nd Law [8] relating force to momentum change with respect to time. Applying this to every lattice point located at the obstacle boundary, $\mathbf{x}_b$, and summing them all up gives the total force, F, exerted on the obstacle by the fluid [9]:

$$\mathbf{F} = \sum_{all x_b}\sum_{\alpha\neq 0}\mathbf{e}_\alpha[f_\alpha(\mathbf{x}_b, t) + f_{\bar{\alpha}}(\mathbf{x}_b + \mathbf{e}_{\bar{\alpha}}\Delta t, t)]. \tag{8}$$

Here $\mathbf{e}_\alpha$ represents velocity in direction α and $\mathbf{e}_{\bar{\alpha}}$ the reflected velocity in the opposite direction, meaning $\mathbf{e}_{\bar{\alpha}} = -\mathbf{e}_\alpha$. Using Eq. 8 we can now extract the **x** component of force (corresponding to car drag) and analyse the effect of varying the car separation distance on this value.

EXPERIMENTAL DETAILS

Before beginning to code complex turbulent flow, the basic techniques of the LBM were implemented in a simpler Python script which could be built upon later. A special effort was made throughout the program to avoid 'for loops' and 'if/else' statements as these can dramatically slow the speed of the simulation by orders of magnitude. Instead the `numpy` library was utilised which manipulates data in entire arrays all at once.

The velocity field $\mathbf{u}(\mathbf{x}, t)$ was first calculated and then used to determine equilibrium distribution of the fluid in Eq. 5. It was now possible to account for collisions and streaming at each time step by including Eq. 6 and Eq. 7 respectively in the Python code.

To incorporate obstacles, the streaming function needed to be modified in order to consider some lattice sites being inside a solid boundary. A boolean mask array was initialised with the same dimensions as the lattice. False values corresponded to empty sites whereas True values were associated with obstacles. For each point on the lattice, the program checked if the mask here was True or False. If True then the distribution was set to 0 as it is not possible for fluid to exist inside the obstacle. However if the mask was False, a further check needed to be run to test if we were located at a lattice site next to the obstacle or if we were far away from the obstacle (done by rolling the mask in the direction the fluid is coming from onto the current site). If located next to the obstacle, then the fluid distribution must have been reflected and therefore has a value equal to the flow in the opposite direction. On the other hand, if the mask is far away then the original streaming step can be run.

This code was applied to an example of poiseuille flow through a pipe to check it was working in the expected manner. This was then extended to simulate turbulent flow. The system was setup in the form of a wind tunnel and the incoming fluid was adjusted to have slightly randomised y-velocities to encourage vortices to form faster.

To create complex shaped masks, a technique was used that could allow a user to draw the obstacle in a binary text file before importing the data into the Python program. 1s and 0s corresponded to True and False values respectively and each digit represented a single lattice point. The Python script read this file and overlayed it onto a custom region on the lattice. The advantage of this method is that the user can see a real time graphic of what they are drawing, allowing easier mask creation as well as extreme versatility for different shapes. Wheels were absent on the model as the 2D nature of the simulation prevents any depth information. The presence of a wheel would be like creating a cylindrical tyre spanning the depth of the vehicle, blocking all airflow underneath the car.

An additional mask for the road was also implemented. Upon an initial run of the experiment, it was noticed that the air nearest the ceiling was experiencing friction and the vortices coming off the back of the car appeared back at the front of the lattice after a large number of time steps. This was because the edges of the lattice were connected, and to fix this the lattice was extended to 3200 sites long and a ceiling added to make the environment more realistic to a wind tunnel.

The experiment was then run with 2 cars present and the force on the back car was recorded using Eq. 8. It was possible to assume the force was proportional to change in momentum as the time-step was 1 and the force would later be normalised to only consider relative force.

Finally, for each run of the program, images were produced to visually examine the motion of the fluid as well as a `.csv` file which held the total drag force on the vehicle at each time step. These could then be imported into Origin for data analysis and graphing of the relationship between car separation and drag.

RESULTS

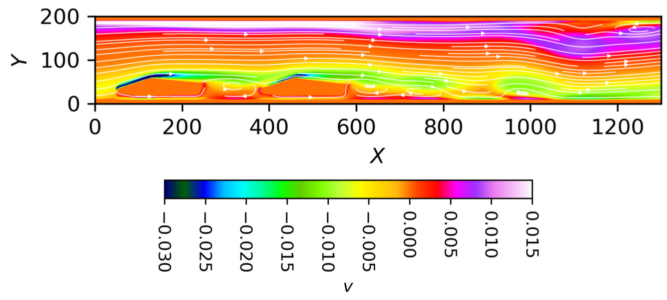

FIG. 1. Image showing vorticity distribution in lattice with overlayed velocity streamlines. Taken after 24 time steps with separation distance 374 cm and Reynolds number of 17280.

Fig. 1 shows turbulent flow as expected from Eq. 2. The image shows the fluid's motion at a singular time step for a given car separation, however similar images were created for every time step at each separation distance. This allowed the turbulent behaviour of the fluid to be examined over time. The difference in oncoming fluid between the front and back car resulted in a disparity in drag which was extracted from all repeats of the simulation and combined onto a single plot:

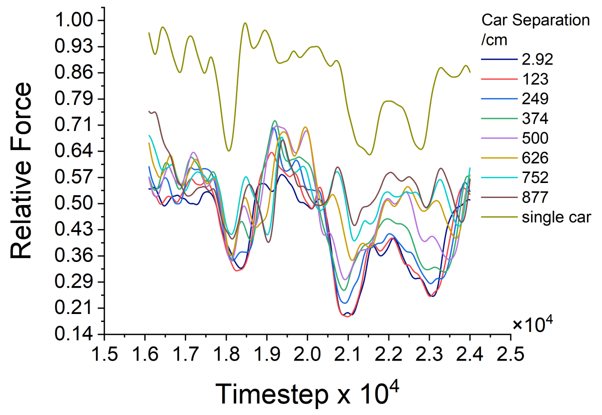

FIG. 2. Graph illustrating the relative force evolution through time for various car separation distances. A repeat was included for a single car without any slipstream (limit of separation $\rightarrow \infty$).

As seen in Fig 2, the single car without any slipstream faces the highest amount of drag. Furthermore, as the 2nd car gets closer the relative force appears to lessen which supports the hypothesis that the slipstream has more of an effect at shielding it from the resistive forces. The first part of the graph had to be removed (time step does not start at 0) as when the simulation was initialised the force was inaccurately high for a few time steps. This skewed the scale, making it hard to see any interesting detail. Furthermore it takes time for the slipstream to reach the car behind meaning the fluid must travel a distance of 1 car length plus the separation distance before any data can be considered. To quantify the effect of the slipstream on drag, a new plot was created to show the relationship between average force and car separation:

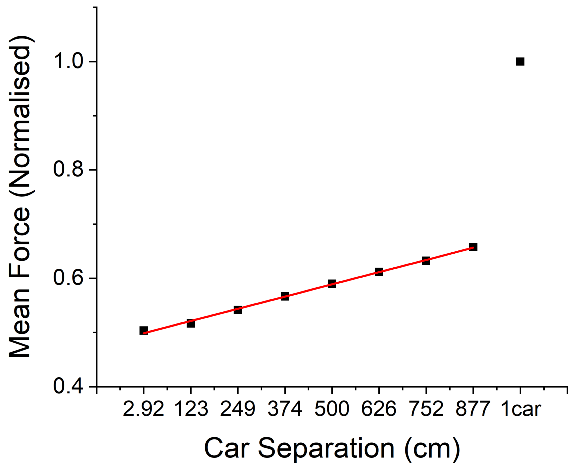

FIG. 3. Graph showing how the relative drag differs for various car separation distances. Linear fit with gradient = 0.0225 and intercept at y = 0.476. Final data point (for a single car without any slipstream) plotted but not included in linear fit.

Fig. 3 clearly shows a positive, linear correlation between mean drag and car separation. This aligns with the observations made in Fig. 2 and proves that slipstreaming effects are dominant at closer proximities. The plot was normalised so all data points were proportional to a car without any slipstream. The lack of error bars on this plot was a consequence of ambiguity surrounding the units used for constants in the simulation. These arbitrary units made it impossible to quantify a numerical error. The intercept value of y = 0.476 is the (proportional) minimum possible air resistance a car can face if it were touching the car in front. The lowest drag simulated was just 54% (at 2.92 cm) compared to that of a car not in any slipstream.

DISCUSSION

In Fig. 2, the order of the line plots matches that of the separation distance (to be expected) in the local region. However there are great disparities in force across the graph as a whole, shown in the form of peaks and troughs. Initially a singular vortex is formed (seen between the cars in Fig. 1) by the leading car which eventually 'hits' the car behind. After this has passed, the same thing happens at later times, leading to rhythmic periods in changing air vorticity and consequently drag. Interestingly, there seems to be a slight x-offset between each line. As mentioned previously, this is because of the extra time it takes for the first vortex to travel to the rear car at greater distances. This had to be accounted for in the measurements of average drag by only considering data after a great enough time step where all the slipstreams had enough time to form.

The Cybertruck was specially chosen for this experiment due to its prism-like design. This simulation was limited to 2D and so only a cross section could be considered. The angular design of the Cybertruck made it the best fit as its dimensions barely vary throughout its depth, allowing it to yield more accurate results than other shaped vehicles.

Overall the linear relationship seen in Fig. 3 agrees with the expectation that drag increases at greater separation distances. All of the car separations experience less drag than without any slipstream (final data point). However upon analysing literature, the graph is incomplete. According to a study which measured drag on a Sedan car behind a Semi trailer-truck [10], the true relationship starts as linear (agreeing with Fig. 3) but then starts to reduce after a critical value, followed by another region of increase. This creates the idea of an air 'pocket' with lower drag further back. To see this effect, our simulation must be repeated for higher separations.

While following by a distance of 2.92 cm yields the lowest resistance of 50.4%, in practice it would be unrealistic to maintain and the risk for collision would be high. Following instead at 123 cm, only adds 1.32% more drag (at 51.7%) while dramatically improving safety.

The Reynolds number used in this experiment was 17280. This is smaller than the value in air: 2.2×10^5 [6]. To increase this number, τ must be set closer to 0.5. However decreasing τ lower than 0.6 led to instability in the program. A further way to improve the accuracy of the investigation would be to use a smaller time step, or a higher resolution lattice. This would improve accuracy of results at the cost of computing speed.

CONCLUSIONS

Overall the investigation achieved its goal of simulating fluid dynamics past a car and measuring the effect of slipstreaming on drag force. Despite the limitations of the LBM, incorrect Reynolds number and the constriction of 2 dimensions, the program successfully found a positive, linear relationship between drag force and car separation inline with the literature. For the Cybertruck vehicle it was discovered that drag could be reduced by 49.6% at the closest measured following distance of 2.92 cm. However it was more practical to increase that distance to 123 cm with only a small sacrifice to drag: now giving a 48.3% reduction. In summary, the findings matched the hypothesis well however lacked the breadth of data points needed to observe the 2nd slipstream 'pocket' located further back.

REFERENCES

[1] *Beginners Guide to Aeronautics: Drag Coefficient*, Nancy Hall, National Aeronautics and Space Administration, (2023).

[2] *How much do we save with the slipstream by riding in the velodrome?*, Iván Herrero, magazine bkool, (2023).

[3] *Aerodynamic drag in cycling pelotons: New insights by CFD simulation and wind tunnel testing*, Bert Blocken, Thijs van Druenen, Yasin Toparlar, Fabio Malizia, Paul Mannion, Thomas Andrianne, Thierry Marchal, Geert-Jan Maas, Jan Diepens, Journal of Wind Engineering and Industrial Aerodynamics. Vol 179, (2018).

[4] *How Mika Hakkinen's famous overtake on Michael Schumacher was repeated by Esteban Ocon*, Connor McDonagh, crash F1 News, (2022).

[5] *Examining Reynolds Number For Turbulent Flow*, Cadence CFD, Cadence System Analysis (2023).

[6] *Nektar++: An open-source spectral/ element framework*, Chris Cantwell, David Moxey, A. Comerford, A. Bolis, G. Rocco, Gianmarco Mengaldo, Daniele De Grazia, Sergey Yakovlev, Jean-Elo Lombard, Dirk Ekelschot, Bastien Jordi, Hui Xu, Yumnah Mohamied, Claes Eskilsson, B. Nelson, P. Vos, Cristian Biotto, Robert Kirby, Spencer Sherwin, Computer Physics Communications 54(**3**), (2015).

[7] *Computational fluid dynamics: Principles and applications*, Jiri Blazek, (2015).

[8] *Classical mechanics. Second ed.*, Tai L Chow, (2013).

[9] *Force Evaluation in the Lattice Boltzmann Method Involving Curved Geometry*, Renwei Mei, Dazhi Yu, and Wei Shyy, University of Florida, Gainesville, Florida. NASA Center for AeroSpace Information. (2002).

[10] *Investigation of Slipstreaming Effect between a Semi trailer-truck and a Sedan Car*, Rubiat Mustak, Md.Habib Ullah Khan, Md.Mahbubur Rahman, International Journal of Scientific & Engineering Research. Vol.8(**2**), (2017).