

Simulating flock of birds using only vision

Žiga Leskovec and Maksimiljan Vojvoda

Collective behaviour course research seminar report

December 17, 2023

Iztok Lebar Bajec | izredni profesor | mentor

When modeling collective behaviour it is commonly assumed that agents inherently know other agents position, velocity and direction. There exists a motivation to get rid of these assumptions and model the behaviour based on how internal and external information is acquired and processed. Vision is one of the more important sensory systems that provide crucial external information, which turns out to be sufficient for modeling interactions between agents in a swarm. In this seminar we explore a mathematical framework for perception-based interactions proposed by Renaud Bastien and Pawel Romanczuk.

Simulation | Vision | Flock of birds

Models of collective behaviour often rely on interactions that do not have a direct physical reality (such as neighbour velocity, relative position and direction). One example of this is the simulation of fish schools [1]. However, this assumption of how the information is processed by agents limits our understanding of the underlying complexity that takes place in such phenomena. A better alternative would be to model the behaviour around internal and external information that agents are capable of acquiring.

In this seminar we use a mathematical framework proposed by Renaud Bastien and Pawel Romanczuk [2], to create a simulation of flock of birds that solely relies on vision.

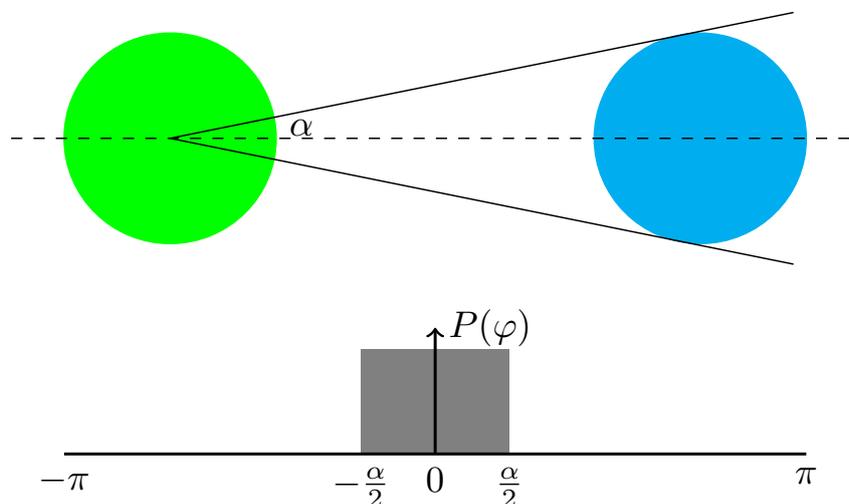
Methods

The simulation is done in 2-dimensional space where agents are represented as simple disk objects with full 360° view. On each simulation step, their velocity is modified based on a projection of their surrounding visual field. This simulates a primitive form of vision.

Visual field projection. Objects around the agent are projected onto their visual field, described by $P(\varphi)$. Function $P(\varphi)$ represents visual obstructions of the agent, where φ is an angle of the visual field. The result is binary, where 0 represents “not obstructed” and 1 represents “obstructed”. An example of a visual field projection can be seen on figure 1.

Velocity. On each simulation step, the velocity of agent i , is modified by Δv_i . The general speed delta is summarized with the following equation:

$$\Delta v_i = F_{\text{ind}}(v_i) + F_{\text{vis}}(P_i) \tag{1}$$



A model of collective behavior based purely on vision

When modeling collective behaviour it is commonly assumed that agents inherently know other agents position, velocity and direction. There exists a motivation to get rid of these assumptions and model the behaviour based on how internal and external information is acquired and processed. Vision is one of the more important sensory systems that provide crucial external information, which turns out to be sufficient for modeling interactions between agents in a swarm. In this seminar we explore a mathematical framework for perception-based interactions proposed by Renaud Bastien and Pawel Romanczuk.

Simulation | Vision | Flock of birds

Figure 1. Projection field graph of a single object.

where F_{ind} function represents speed delta collected from **individuals** “internal information”:

$$F_{\text{ind}} = \gamma(v_{\text{pref}} - v_i)\hat{v}_i \quad [2]$$

where γ represents speed relaxation rate, v_{pref} **preferred** speed of the individual and \hat{v}_i normalized direction vector. Function F_{vis} transforms **visual** field to the individuals speed delta. It is independent of other individuals properties, and is described with the following equation:

$$F_{\text{vis}}(P) = \int_{-\pi}^{\pi} G(P, \varphi) h(\varphi) d\varphi \quad [3]$$

Here $G(P, \varphi)$ encodes how information from the visual field impacts the movement, while $h(\varphi)$ encodes properties of the perception-motor system, in our case, it describes how front-back distance impacts the speed and how left-right distance influences the heading direction of an agent. For convenience the equation is split into two parts

$$\Delta v_i = \gamma(v_{\text{pref}} - v_i) + \int_{-\pi}^{\pi} \cos(\varphi) \alpha_0 (-P_i(\varphi) + \alpha_1(\partial_{\varphi} P_i(\varphi))^2) d\varphi \quad [4]$$

$$\Delta \Psi_i = \int_{-\pi}^{\pi} \sin(\varphi) \beta_0 (-P_i(\varphi) + \beta_1(\partial_{\varphi} P_i(\varphi))^2) d\varphi \quad [5]$$

The first (Δv_i) describes speed delta, while the second ($\Delta \Psi_i$) describes heading angle delta. Consequently the heading vector is now removed, since it is encoded as the heading angle.

As we will see further in the seminar, parameters α_1 and β_1 influence the equilibrium distances. α_1 influences the front-back distance equilibrium $\frac{r}{\alpha_1}$, while β_1 influences the left-right distance equilibrium $\frac{r}{\beta_1}$, where r stands for agent radius.

Integration of P. One of the core elements of the simulation is calculating integrals of functions multiplied with P such as $\int_{-\pi}^{\pi} \cos(\varphi)P_i(\varphi)$ from equations 5 and 4. Because the projection function P contains contiguous regions of values 0 or 1, we can split the integral into multiple definite integrals where we calculate only the sections where P has value of 1. A visual representation of multiplying a trigonometric function with P (in this case \cos) can be seen on figure 2. This trick trivialises the integral calculation.

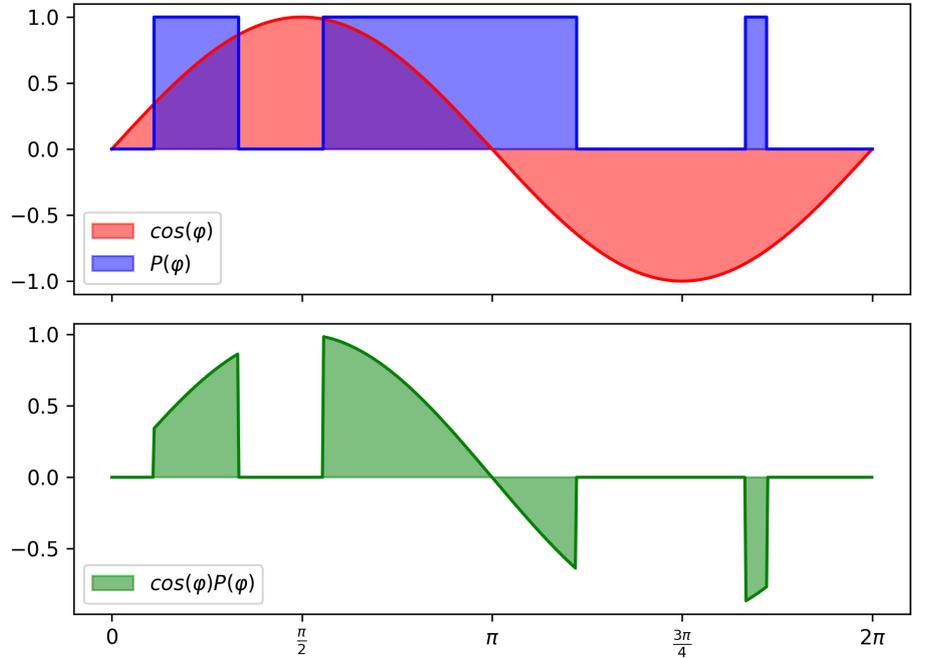


Figure 2. Multiplying a function (in this case \cos) with some projection function P .

Implementation. The simulation is implemented in C++.

In the source paper [2], the author calculates the projection function P using ray casting. We have improved on this by projecting the objects directly by calculating the appropriate angles of vision for each object as show in figure 1. For each object we constructed an array of intervals where P would have a value of 1. By carefully merging these intervals we kept the array *clean* as none of the intervals would intersect. With this array we could trivially calculate all the necessary components described in equations 4 and 5 using the method described in the previous section.

The underlying simulation loop is also parallelizable, as agents are dependant only on the previous state of the simulation, therefore we can use multi-threading to increase the simulation speed, thus allowing us to simulate larger or longer simulations.

Simulation visualisation. The speed of image generation is not important to this seminar and is therefore done separately in Python with Matplotlib. Every simulation figure has its own one minute long animation available on our Github repository <https://github.com/siggsy/collective-vision/tree/main/results>. It is worth noting that the size of the agents is not to scale.

Results

Every simulation was run for 2000 steps with 50 agents randomly placed in a box bounded by (0,0) and (5,5). All agents had a radius of 0.5 and had v_{pref} set to 0.5. Simulation parameters were set to the following values:

- $\alpha_1 = 0.08$
- $\beta_1 = 0.08$
- $\gamma = 0.95$

In figure 3 we can see 3 different collective behaviours using α_0 and β_0 parameters found in the paper this seminar is based on[2].

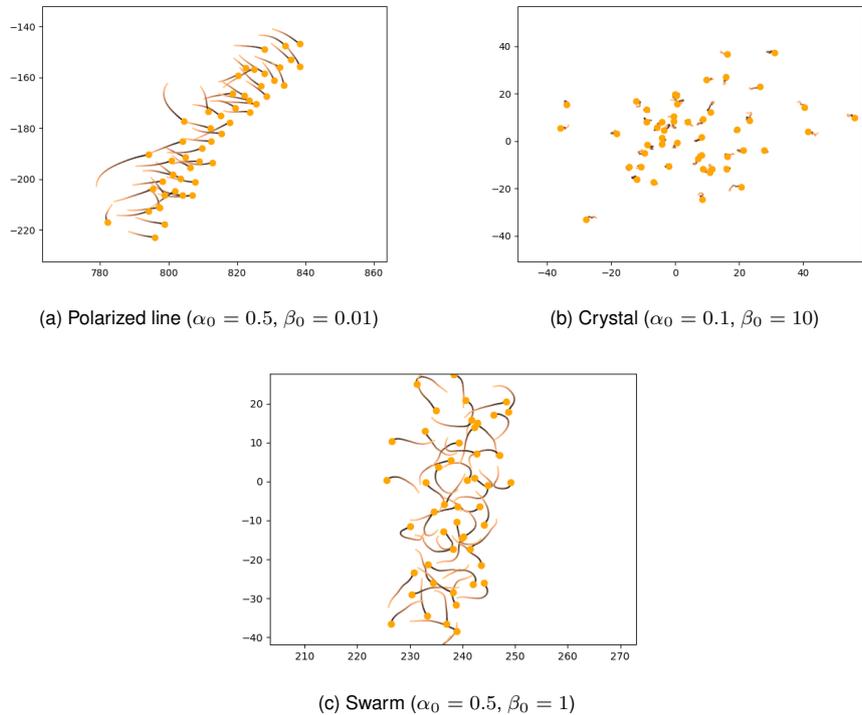


Figure 3. Last frames of the tested simulations

In simulation (a) all agents move in one general direction and form a line. In contrast, in simulation (b) agents shake in-place and barely make any distance. Since the figures represent the last frames of the animation, this can be observed by checking their final coordinates. In the final simulation (c) agents are moving around freely while maintaining a swarm.

Discussion

In this seminar we implemented and simulated a collective behaviour of birds using only their vision. We have improved on the original work by Bastien and Romanczuk [2] by replacing ray casting with direct projection. This improved the accuracy and calculation efficiency of the visual fields of the birds.

The simulation is limited to only 2 dimensions as visualisation of more would be quite cumbersome.

CONTRIBUTIONS. Žiga Leskovec fine tuned the simulation parameters and handled the simulation visualisation; Maksimiljan Vojvoda implemented the simulation loop.

Bibliography

1. Huth A, Wissel C (1992) The simulation of the movement of fish schools. *Journal of Theoretical Biology* 156(3):365–385.
2. Bastien R, Romanczuk P (2020) A model of collective behavior based purely on vision. *Science Advances* 6(6):eaay0792.