Exploring Swarm Behavior: An Undergraduate Project in Physics and Computer Programming

Dr. Guenter Bischof, Joanneum University of Applied Sciences

Guenter Bischof is currently an associate professor at Joanneum University of Applied Sciences and teaches engineering and applied mathematics.

Mr. Konrad Dobetsberger, Joanneum University of Applied Sciences

Konrad Dobetsberger is currently studying Automotive Engineering at Joanneum University of Applied Sciences. Prior to attending university, he completed his education at a higher technical education institute with a major in mechatronics.

Mr. Markus Ensbacher, Joanneum University of Applied Sciences

Markus Ensbacher is currently studying Automotive Engineering at Joanneum University of Applied Sciences. His research interests encompass internal combustion engines, drivetrain technologies, and chassis development. Prior to attending university, he completed his education at a higher technical education institute with a major in mechanical engineering.

Mr. Christian J. Steinmann,

Christian Steinmann has an engineer degree in mathematics from the Technical University Graz, where he focused on software quality and software development process assessment and improvement. He is manager of HM&S IT-Consulting and provides services for S

Mr. Alexander Strutzenberger, Joanneum University of Applied Sciences

Alexander Strutzenberger is currently studying Automotive Engineering at Joanneum University of Applied Sciences. Prior to his studies he visited a secondary school with a focus on natural sciences.

Exploring Swarm Behavior: An Undergraduate Project in Physics and Computer Programming

Günter Bischof [1], Konrad Dobetsberger [1], Markus Ensbacher [1], Christian J. Steinmann [1, 2], and Alexander Strutzenberger [1]

[1] Joanneum University of Applied Sciences, Institute of Automotive Engineering, Graz, Austria
[2] HM&S IT Consulting, Graz, Austria

Abstract – Collective motion, epitomized by the fascinating spectacle of birds flocking in unison or fish moving cohesively through the depths of the ocean, has captivated the human imagination for centuries. Beyond its aesthetic allure, this natural phenomenon demonstrates self-organization, where order arises from disorder without centralized control. This self-organizing principle finds equivalents in physical phenomena such as phase transitions and spontaneous symmetry breaking and has also significant implications in biology, chemistry, robotics, and even social sciences.

As part of an interdisciplinary undergraduate student project at the intersection of physics, mathematics and computer programming, student teams were given the opportunity to immerse themselves into the world of self-organization and swarm behavior. Two models, the Vicsek and the *boids* model, were used to simulate swarm behavior.

The Vicsek model is a simple, mathematically rigorous approach rooted in statistical physics, while the *boids* model emphasizes the behavioral aspect of collective motion, making it suitable for creating realistic animations and simulations of swarm behavior. In addition, it can be extended to include obstacles and environmental factors that affect the swarm behavior.

The task of our students was to develop − as a team of three − a computer program in C#, in which both models are implemented and visualized. Teamwork was an additional challenge, as organizational skills were required in addition to the underlying task, such as holding meetings with collaborative decision-making, and an appropriate division of labor in the development of the software.

The resulting computer program with an intuitively designed user interface allows the simulation of different scenarios due to a variety of adjustable parameters. The visual output of the program reflects the different model assumptions and thus promotes the understanding of model building in general and of self-organization and swarm behavior in particular. The program is freely available and can be downloaded from our institution's home page.

Introduction

Swarm behavior, often exemplified by the coordinated movement of birds or fish, has long captivated the fascination of scientists, engineers, and nature enthusiasts alike. The collective intelligence displayed by these groups of relatively simple individuals, when organized into a cohesive unit, is a testimony to the marvels of nature (Figure 1). Understanding and simulating this phenomenon has recently come into focus of research in various fields, from biology and biophysics to computer science and robotics.

Figure 1: Snapshot of a flock of birds

In search of a fundamental theory of life, physicists have been interested in the study of "active matter", which ranges from swarming molecules to schools of fish and flocks of birds, for quite some time. One of the first groundbreaking experiments for the investigation of active matter was conducted by Stanislas Leibler in the mid-1990s [1], [2]. His research group was able to show how complex, life-like structures can form from microtubules and a few proteins supplied with adenosine triphosphate (see, e.g., [2] and [3]). Around the same time, Tamás Vicsek developed an influential model of active matter. He had been working on the collective motions of bacterial colonies and of flocks of birds but saw no adequate explanation by existing theories. As a new approach, he tried the model of ferromagnetism according to Werner Heisenberg, in which each atom can be regarded as a freely rotating bar magnet. In this model, large-scale magnetism emerges when a majority of the atomic magnets align due to their interaction. For the modelling of active matter, Vicsek replaced the small magnets with moving arrows symbolizing the individual particles of the colonies or flocks [1]. The velocity of each individual particle aligns itself in the model with the average velocity of its neighbors, but with a certain amount of randomized error. This concept ultimately led to Vicsek's model for the study of active matter [4]. Once enough arrows were packed, they moved after some time in patterns similar to those of flocks of birds and fish, at least according to his simulations.

At the same time, there were developments in the field of computer animation, which were driven by the computer graphics community. Almost a decade before the publication of the Vicsek model, Craig Reynolds developed a groundbreaking artificial life program for the simulation of birds. He referred to the simulated objects as bird-like, "bird-oid" objects (*boids*) [5]. Since then, the *boids* model has been widely used for computer graphics that provide a realistic depiction of birds or other living creatures, for example in the 1998 video game Half-Life by Valve Corporation. Like the Vicsek model, the *boids* model exhibits emergent behavior., i.e., the complexity of the model results from the interaction of the individual agents (the *boids*) that follow a simple set of rules. The modeler can define how the individuals will respond to local events or conditions. The global behavior of the swarm is, thus, an emergent phenomenon, as none of the rules for the individual *boids* depend on global information.

The model's minimum set of rules includes the following behavioral tendencies: move toward the perceived center of mass of the *boids* within the visual range, match velocities with the other *boids* within this range, and maintain a minimum distance from other objects. These simple rules can lead to complex, seemingly concerted behavior on the part of the swarm. They constitute a kind of generalized genotype of the *boids* system. The resulting flocking behavior can thus be regarded as the generalized phenotype of the system. It is related to its genotype in the same way as the morphological phenotype of an organism relates to its molecular genotype [6].

Both the Vicsek and *boids* models rely on a manageable set of rules, which can lead to very complex system behavior. In addition, such systems can be observed in nature; it is therefore possible to identify suitable model parameters by observing flocks of birds or other creatures, such as schools of fish or herds of animals. These are ideal prerequisites for using these models as project assignments for student projects in the field of modelling and computer programming.

In the past academic year, this assignment has been added to the list of proposals for our interdisciplinary undergraduate student research projects. The basic idea as well as the design of these projects, the implementation of this particular project by a student team, and the results as well as the learning experiences of the students in the course of this project are the subject of this paper.

Interdisciplinary undergraduate student research projects

Over the past two decades, we have established a team-oriented project-based learning environment at our institute to support our students in linking the skills and knowledge taught in the various basic subjects of a typical undergraduate engineering degree program.

Computer programming is the ideal hub for this learning environment because modern engineering increasingly takes place in a virtual world, and we therefore have to teach students – even in the classic engineering disciplines – the necessary programming skills. The starting point is the software development course in the first year of study. In this course the programming language C# is introduced, an object-oriented cross-platform programming language of the C family that facilitates the development of graphical user interfaces.

The first phase of our project-oriented teaching comprises the second semester. At the beginning of the semester, the students are offered – complementary to the computer programming course – a variety of project proposals. The project tasks are defined in such a way that students are requested to apply already existing or still to be acquired knowledge in the STEM subjects to appropriate problems. The fact that students have the opportunity to choose a topic from a list of project proposals increases the acceptance of the assignment and promotes motivation to work on it. A team of three of four students works on such a project, with one of them taking on the role of team leader. This structure promotes the development of generic skills, like the ability to work in teams, to meet deadlines and to keep records. If the students' selection of project tasks allows, two or three groups are assigned with the same task in order to generate competition, which evidently increases the students' motivation (see, e.g., [7] and [8]).

For the approximately 90 participating students, a corresponding number of project tasks must therefore be found each year. The role of the project supervisors is to guide the students through the entire process and give them the necessary support to succeed. To this end,

meetings are scheduled if necessary or additional lectures are offered in order to impart knowledge and skills needed and not yet available for the project in a timely manner. The result of each project is a stand-alone computer program that runs on Windows operating systems and fulfills the required project task. At the end of the semester, each team presents its solution to the entire class and submits all required project documents (technical report, minutes of meetings and an instruction manual for the program).

The project task, which is described in this paper, was the creation of a computer program that simulates and visualizes the dynamics of a flock of birds as realistically as possible. As a first step, a swarm based on the Vicsek model should be generated. Particular attention should be paid to the visualization of the individual particles, which were to be displayed as small arrows. In addition, the user should be given the option to define all parameters of this model as desired. This involves the number of particles, their radius of interaction and the degree of randomized perturbation of the flight direction of the individuals. In a further step, the *boids* model has been realized, which has more adjustable parameters, and thus also offers a wider range of swarm behavior. This model also allows the integration of predatory elements, such as birds of prey, into the simulation.

The project work started with an introductory lecture for the project participants, in which the underlying model assumptions for the simulation of swarming, in particular those of the Vicsek model, were discussed.

Vicsek model

The Vicsek model [4] is probably the simplest model that generates a transition from a disordered to a collective motion. However, despite its simplicity, it is able to describe universal characteristics of flocking behavior.

It is based on the assumption that all individuals move at a constant speed and orient themselves towards their neighbors. Random noise is taken into account, which disturbs the orientation in order to prevent the individuals from aligning themselves too quickly in one direction. With a high density of individuals or low noise, the model quickly shows collective motion.

The Vicsek model is controlled by three parameters: the density of individuals, the amplitude of noise, and the radius R in which the individuals can perceive their neighbors (the visual range, represented by the blue circle in Figure 2).

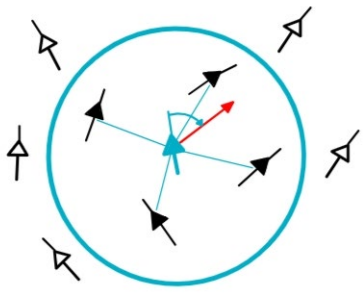


Figure 2: Visualization of the computational rules for the Vicsek model

The model consists of N moving particles, indexed by $i = 1, \ldots, N$, representing the individuals and moving continuously (off-lattice) on the plane. Since the investigation of the

swarm behavior was limited to motions in the plane within the framework of this student project, the direction of motion for each particle can be represented by only one angle θ_i. All particles move with the same speed v. The particles interact with each other within an interaction radius R, which forces them to change their directions.

The velocities of the particles are determined simultaneously at each time step n (step size Δt) and the position of the particle $\boldsymbol{r}_i$ is updated as

$$\boldsymbol{r}_i^{n+1} = \boldsymbol{r}_i^n + \boldsymbol{v}_i^n \Delta t \tag{1}$$

according to its particle velocity $\boldsymbol{v}_i$ (blue arrow in Figure 2)

$$\boldsymbol{v}_i^n = v \begin{pmatrix} \cos(\theta_i^n) \\ \sin(\theta_i^n) \end{pmatrix}. \tag{2}$$

Essential for the dynamics of the Vicsek model is the way how the angles θ_i are updated within each time step. It takes place according to the rule

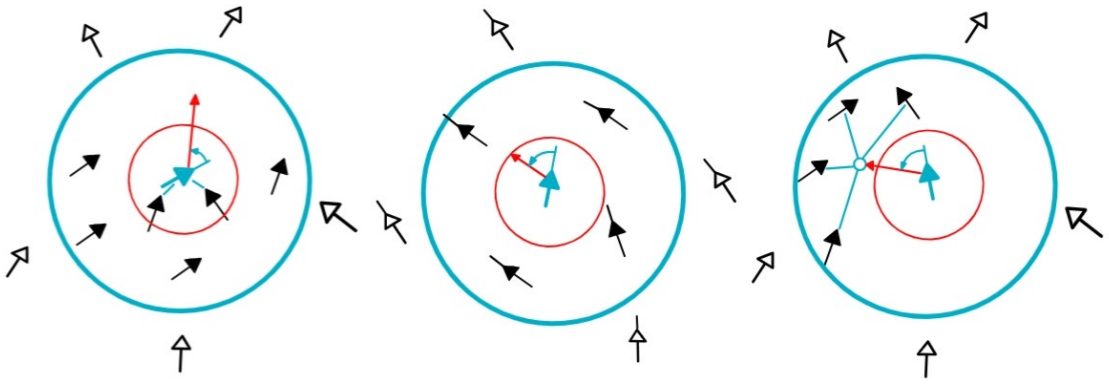
$$\theta_i^{n+1} = \langle \theta^n \rangle_{i,R} + \eta_i^n, \tag{3}$$

where $\langle \theta^n \rangle_{i,R}$ denotes the average angle of all the particles within the interaction radius R (red arrow in Figure 2). The second term, η_i, is a random perturbation obtained from a uniform probability density distribution [$-\eta/2$, $\eta/2$]. Thus, the term η_i^n represents noise, which can be regarded as a temperature-like variable if the swarm is considered as a physical system. Due to this algorithm, the individual particles consume energy and are therefore not in thermal equilibrium. However, this can give rise to a large-scale ordered motion.

For the Vicsek model, periodic boundary conditions were implemented in this project to minimize finite size effects due to finite boundaries. If a particle moves out of the simulation area, it reappears on the opposite side. The interaction radius R is also mirrored on the opposite side, which is why particles at the other end of the simulation area also affect the behavior of the particle approaching the boundary [9].

Boids model

The *boids* model, created by Craig Reynolds [5], is based on three basic rules of behavior that are assigned to each individuum in the swarm: separation, alignment, and cohesion. These are shown schematically in Figures 3 (a) to (c).



(a) Separation (b) Alignment (c) Cohesion

Figure 3: Visualization of the computational rules for the *boids* model

The rule of separation states that each individual *boid* wants to maintain a minimum distance from their neighbors – the protected range, represented by the inner red circle - to avoid

collisions. Each individuum changes direction in such a way that a minimum distance is always maintained from the neighbors (Figure 3 a).

The rule of alignment states that each individual *boid* strives to align in the same direction as their neighbors. It looks at the average direction of the neighbors who are within a certain radius – the visual range, represented by the outer, blue circle – and, if necessary, changes its orientation. As a result, alignment in the swarm is achieved (Figure 3 b).

The rule of cohesion states that each individual *boid* tends to steer to the center of mass of its neighbors in order to keep the swarm together. It calculates the average location of its neighbors within the visual range and moves in its direction to maintain the cohesive structure of the swarm (Figure 3 c).

By combining these three rules, the *boids* model can generate realistic swarm behavior. Each individuum continuously adjusts its motion based on the positions and velocities of its neighbors. This creates a collective behavior in which the *boids* interact with each other to create complex patterns and formations, similar to those observed in nature.

These rules are also extendable; a few more rules have been added to this list in the students' computer program. In particular, repulsive boundary conditions were implemented in this project. In case an individual approaches the edge of the visible region, it reverts to return to the simulation area. In this way, the *boids* behave as if they were flying in an enclosed space. The distance from the edge of the visible region at which the individuals begin to turn around can be adjusted as desired, as well as the degree of their readiness for this change of direction [10]. In addition, the *boids'* speed is constrained to within a configurable range, since real birds cannot fall below a minimum speed in flight and, on the other hand, cannot reach arbitrarily high flight speeds. And finally, a "predator" has been added that can be moved around by mouse drag and must be avoided by all the *boids*.

Implementation into a computer program

The computer program for the simulation of swarm behavior was written by the student project team in C# within the object-oriented software development environment Visual Studio. The program's graphical user interface allows the user to switch back and forth between three program windows, namely Settings, Simulation, and Help (see Figs. 4, 6, 7, and 10).

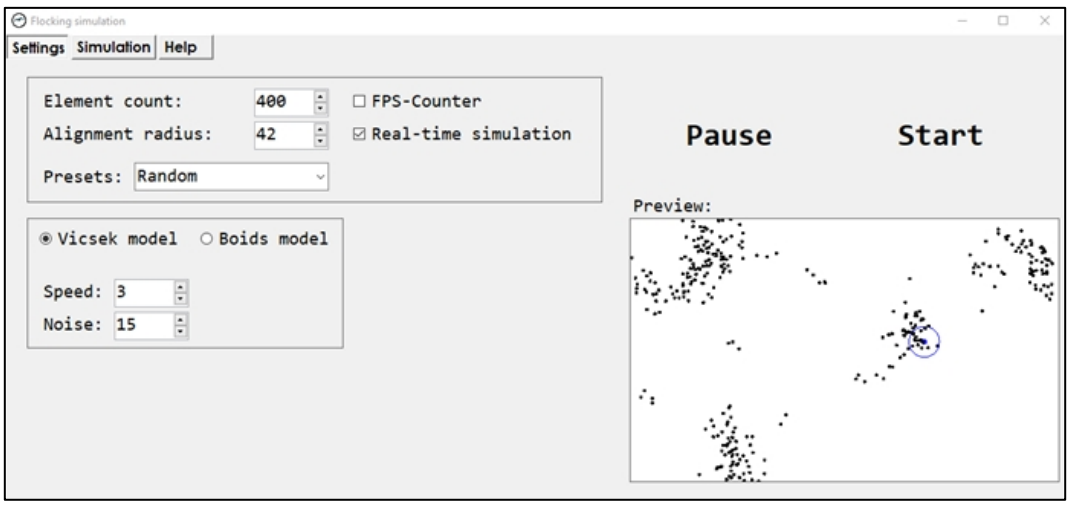


Figure 4: Settings for the Vicsek model

In the Settings tab all model-relevant parameters can be adjusted by the user. An ongoing simulation continues to run in the background when switching to this tab. After starting the simulation in the Settings tab, the program automatically jumps to the Simulation tab. The individual elements are represented by arrows in both the Vicsek and the *boids* model, whereby the representation of the two models differs from each other in terms of color. The Help tab acts as a built-in user guide, which explains the use of the program and the individual settings options.

In the following framed area, there are three input fields and two checkboxes, which both simulation models have in common. In the *Element count* box, the number of elements (particles or *boids*) is entered. The *Alignment radius* specifies the visual range of the particle. The larger the value, the more coherent the resulting swarm structures become. It appears as a blue circle in the preview window (Fig. 4). The *Presets* input field allows to choose either randomly generated initial positions of the elements, or some predefined arrangements.

When the *FPS-Counter* checkbox is activated, a rectangular display will be added to the upper left side of the screen in the simulation tab, which shows the current frame rate of the animation (see Fig. 5).

When the *Real-time simulation* setting is enabled, the simulation runs in real time. This means that as soon as the calculation of the program is completed, the result is immediately displayed as a new image in the animation. This enables high frame rates for a lower number of particles. For higher numbers, on the other hand, it is advisable to switch off real-time simulation. If real-time simulation is deactivated, the simulation runs with fixed time steps that can be specified by the user.

The *Preview* window displays the current simulation parameters and their variation. This preview refers to geometric conditions such as radii, particle counts, the predatory element (if any), and the range boundaries.

After the desired parameters have been entered, the initialization of the simulation is carried out by pressing the *Start* button; the program jumps to the Simulation tab and the animation begins. The simulation stops when the *Pause* button is clicked, and only the last generated image is displayed. By clicking again, the simulation will continue. In the simulation tab, the program can be stopped/started by pressing the space bar.

Swarm simulation with the Vicsek model

To perform the swarm simulation with the Vicsek model, the corresponding radio button must first be selected. Then the associated input field appears, in which the specific simulation parameters can be set (see Fig. 4). In the *Speed* input field, the particle speed can be selected, which remains constant for all generated particles throughout the simulation. The value entered in the *Noise* input field defines the quantity η in equation (3). Large values for η lead to slower swarm formation, or in extreme cases to no flocking at all.

Once the desired parameters have been entered, the simulation is initialized by pressing the *Start* button. This starts a timer that acts as a loop and triggers the computation steps. In the first part of the loop, the particles within the range of view of the particle under consideration are determined. Based on this information, the new positions for each particle are calculated. In the last step of the loop the particle positions are updated according to equation (1). In the Simulation mode, the particles are drawn as arrows (see Fig. 5) and move across the screen at a speed corresponding to the perception of birds' flight behavior.

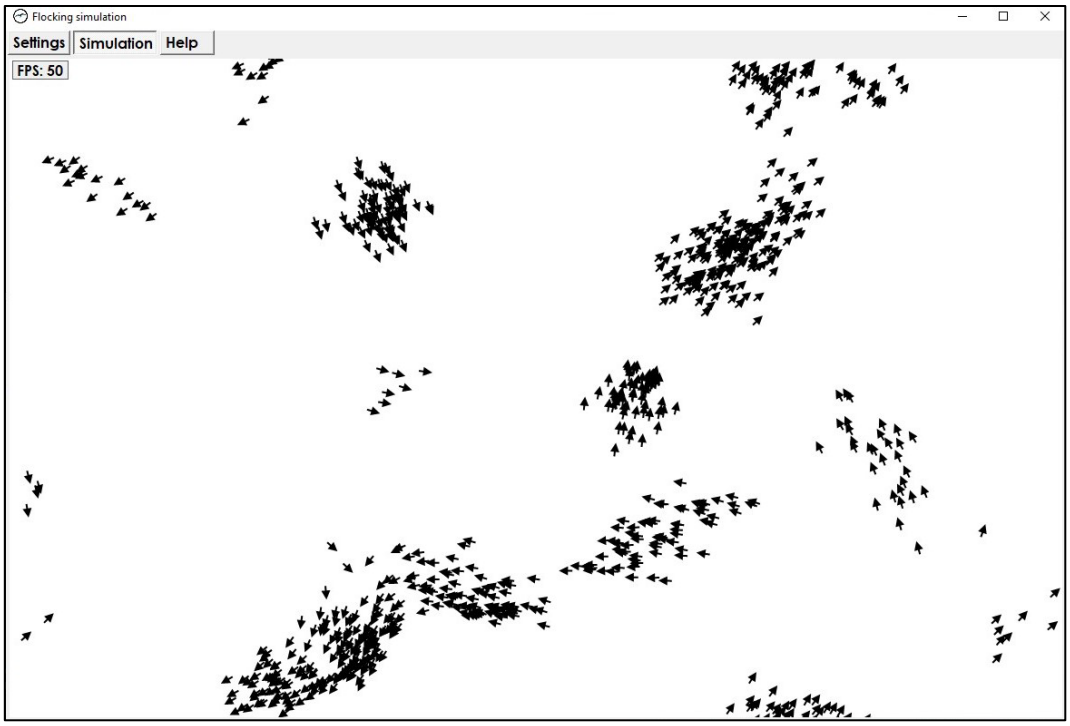


Figure 5: Simulation of swarm behavior in the Vicsek model

Swarm simulation with the *boids* model

To perform the swarm simulation with the *boids* model, the corresponding radio button must first be selected. Then the associated input field appears, in which the specific simulation parameters can be set (see Figures 6 and 8). The *Separation radius* defines the protected range (see Fig. 3 (a)) and determines how close the individual elements may come to each other. This sets the minimum distance between the *boids* in the swarm. A smaller *Separation radius* results in a denser swarm, while a larger radius delays or prevents flocking. The *Maximum velocity* and *Minimum velocity* input fields allow the definition of upper and lower limits for the speed of the *boids*, since in this model, in contrast to the Vicsek model, the speed of the individuals is not kept constant. The *Separation factor* indicates the extent to which the *boids* avoid other elements being within their protected range. A higher numerical value for the *Separation factor*, combined with a large *Separation radius*, results in a swarm with a lower density. The *Cohesion factor* indicates how much the elements are trying to move to the center of the swarm. A higher numerical value for the *Cohesion factor*, in combination with a high visual range, leads to a swarm with a high degree of cohesion. The *Alignment factor* determines how strongly the elements try to take the direction of their neighbors.

The *Borders* input field specifies the distance from the edge of the display window in pixels, where the *boids* begin to change their direction of motion so as not to disappear from the screen. The range boundary is represented by a rectangle in the preview window, giving an overview of the area within which the *boids* move freely (see Fig. 6). A smaller value can cause the *boids* to disappear from the screen for a short time during their reversal maneuver. A large value, on the other hand, restricts the maneuvering area available to the swarm.

The *Turn around factor* determines how strongly and quickly the elements turn over at the range boundary. A combination of a small range boundary and a small *Turn around factor* can cause the swarm to temporarily leave the screen during the turn-around maneuver.

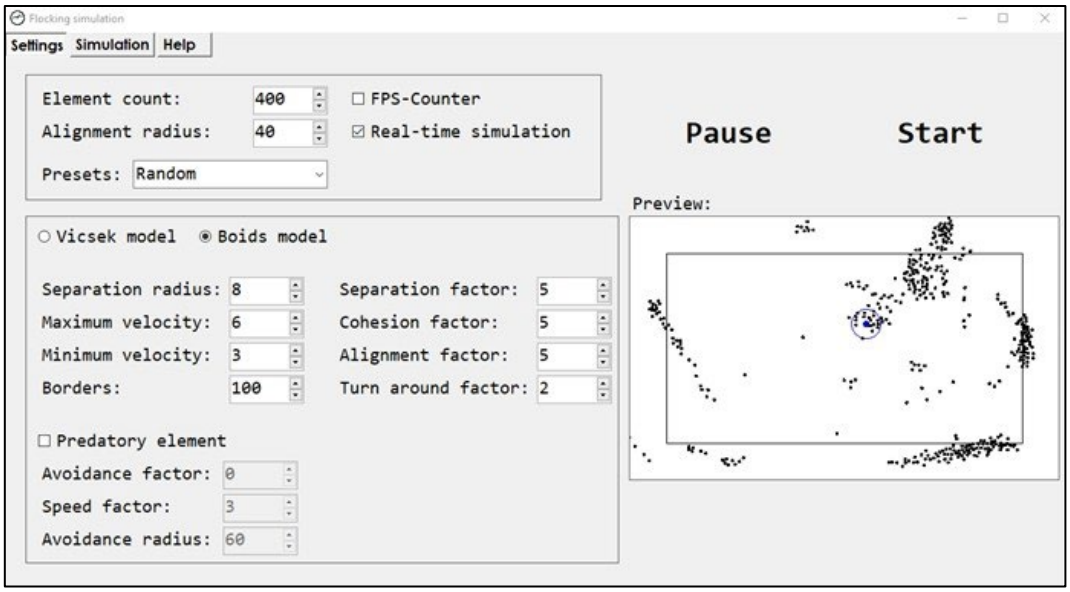


Figure 6: Settings for the *boids* model

In the Simulation mode, the *boids* are drawn as arrows and their color and size are calculated or changed depending on their speed (Fig. 7). Slower *boids* are shown as blue arrows, as they speed up, they change their color to red and increase in size.

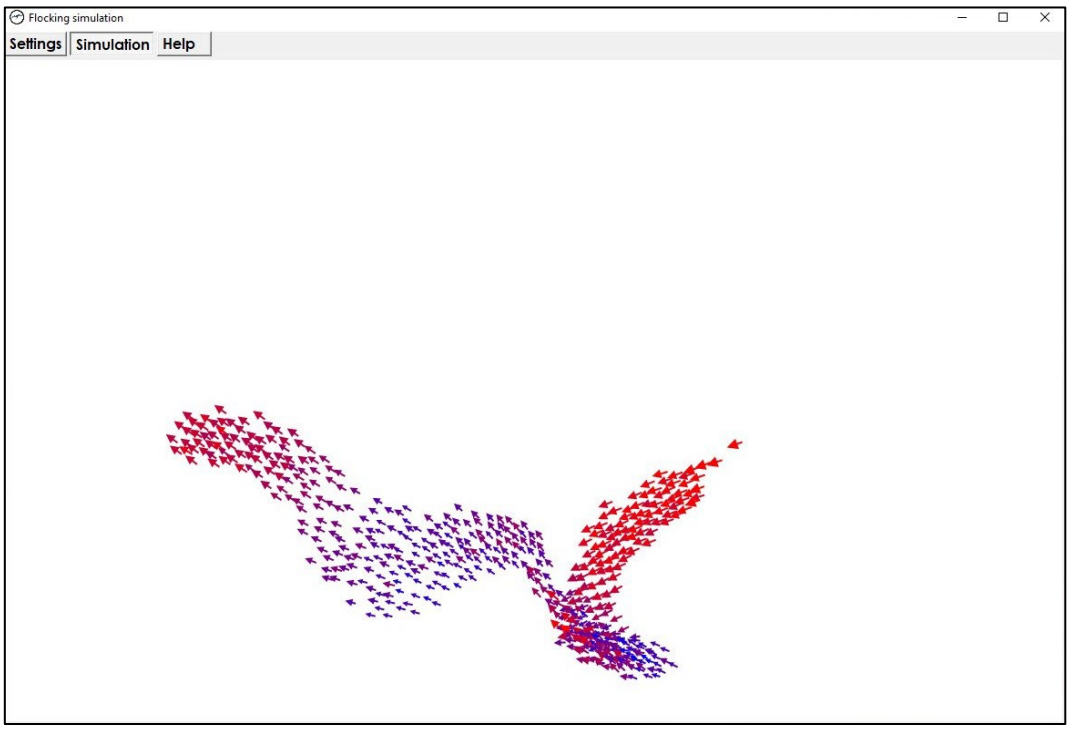


Figure 7: Simulation of swarm behavior in the *boids* model

The *Predatory element* checkbox introduces an element into the simulation that acts like a bird of prey on the swarm and can be controlled by mouse movement. The elements of the swarm try to avoid the predatory element and dodge. The *Avoidance factor* indicates how much the elements of the swarm try to flee from the predator. A larger value leads to a greater repulsion of the predatory element. The *Speed factor* controls how much faster the predatory

element is than the elements of the swarm. For example, a factor of 2 means that the predatory element is 1.2 times faster than the *boids*. The *Avoidance radius* defines the distance at which the predator element is detected from *boids* (red circle in the preview window in Fig. 8). Together with the Avoidance factor, this setting determines how strongly the *boids* try to stay away from the predatory element.

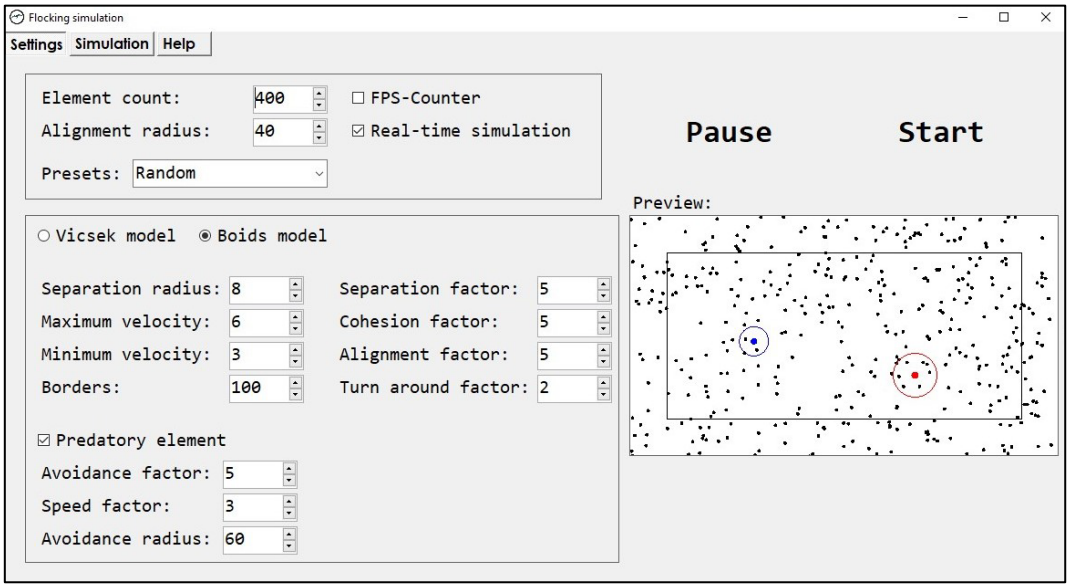


Figure 8: Settings for the *boids* model with predatory element

In the Simulation mode, the predatory element is represented by a circle when it is at rest (see Fig. 9, center). In motion, it is represented by an arrow.

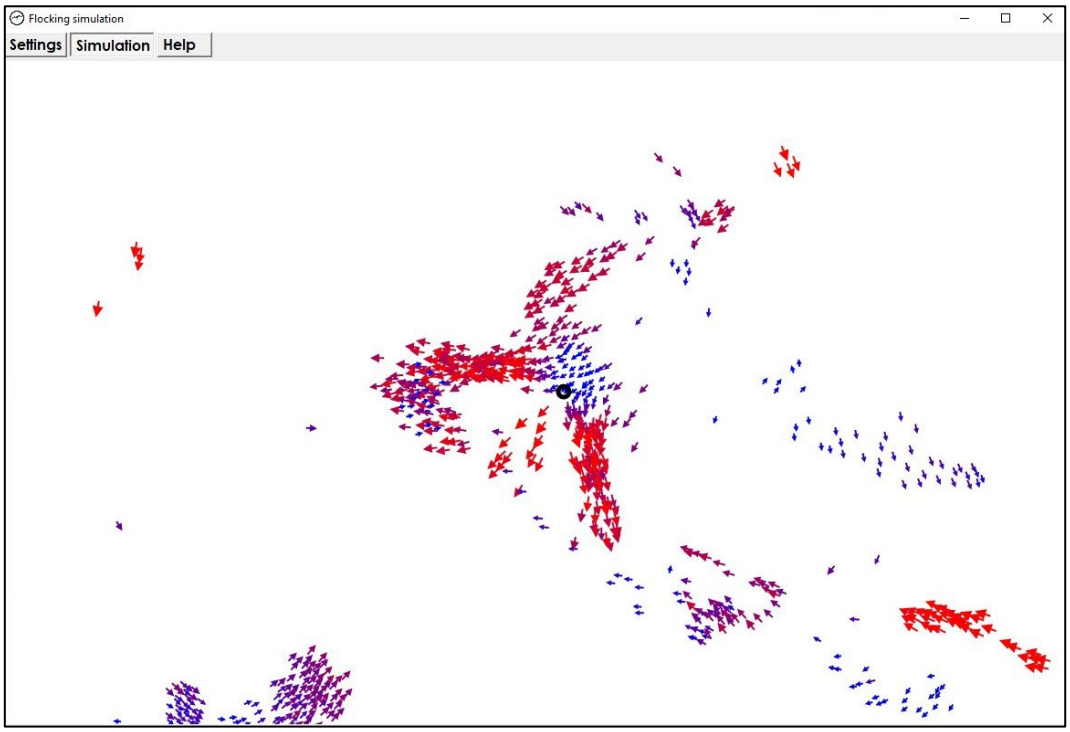


Figure 9: Simulation of swarm behavior in the *boids* model with a predatory element

When the predatory element approaches the flock of birds, or vice versa, the *boids* move away, causing the swarm to split (see Figure 9). When the swarm has broken up, the subswarms reorganize around their own, now distinct and isolated, centers of mass.

In order to provide assistance to new or inexperienced users, a help function has also been integrated into the computer program. In it, all input options are presented, and their effect is subsequently explained. To make orientation as easy as possible, the user interface is displayed, in which the individual input options are numbered consecutively, and the explanation of the respective input parameters takes place based on this numbering (Fig. 10).

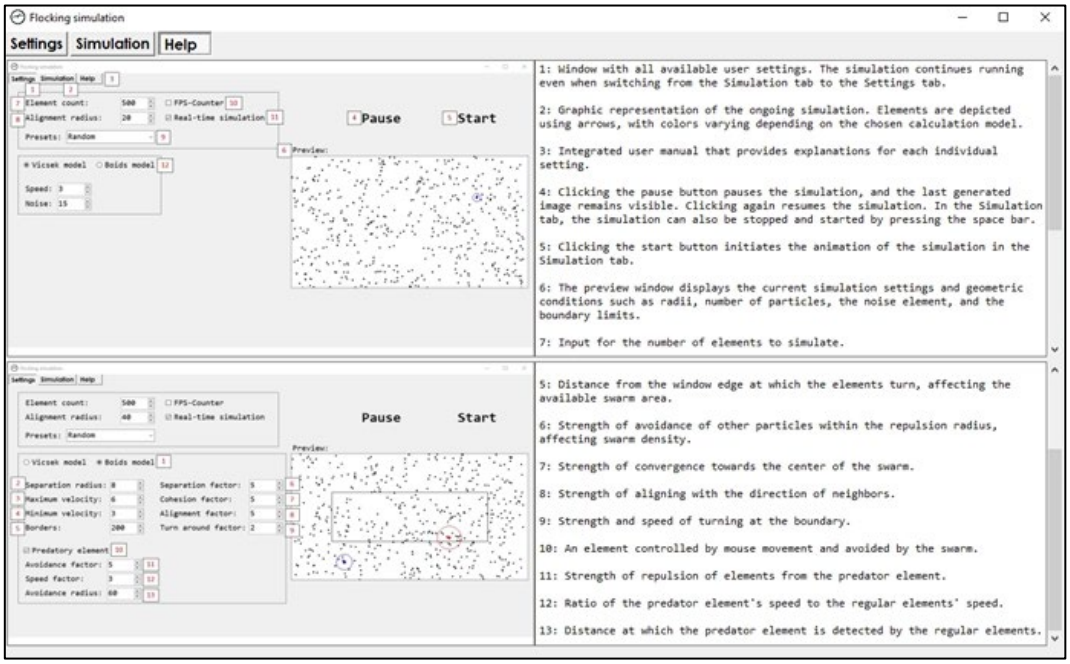


Figure 10: Help function of the simulation program

This integrated help function, which can also be considered as an instruction manual, and the intuitive user guidance make the program easy to use.

Project progression and learning experience

The starting point of the project work after project selection and team building was the kick-off meeting, where the students were familiarized with the Vicsek model by their project supervisors. During a brainstorming session, extended parameter sets, such as those used in the *boids* model, were also discussed. With initial ideas in mind, the students began to structure the task, divide it into work packages and assign them to the individual project members. The students also created a WhatsApp group, which was used for consultations and appointments, as well as an MS Teams project group, on which all data and intermediate results of the program were uploaded. In addition, a Discord group was used for communication during the programming work. In order to assess the progress of the project, the students were required to write interim reports and send them to their supervisors. In subsequent meetings, these reports were used to address problems and develop solution strategies. Of course, the students could also contact their supervisors at any time.

According to the students involved, the programming project has provided valuable new insights. These range from organizational points, such as holding meetings and coordinating collaborative work on a program, up to decisions to be made as a team. It was a surprising realization for the students that even in such a small team, it can be difficult to agree on times for meetings or joint work. Making joint decisions was also a hurdle at the beginning, as compromises often had to be found. In addition, it turned out that it is worthwhile to invest time in careful analysis and good planning, as well as in good documentation and extensive annotation of the program code.

Gaining knowledge through joint problem analysis was also part of the learning experience. For example, in the initial phases of programming, the Vicsek model developed an inexplicable preferential direction to the left. During the analyses, it turned out that the straightforward calculation of the new angles of the particles according to equations (2) and (3) was the cause of that effect. When computing the angles of the next timestep, it was initially overlooked that the averaging of two particles in the visual range of, e.g., 5° and 355° results in the angle 180°, and not the desired angle 0°. In the course of this project, the students learned in a very practical way that a good code review only works well in a team, since it is difficult to discover the own programming errors.

Students were also encouraged to critically reflect on their own work. On the one hand, they presented their results to the entire class at the end of the semester, focusing on the functionality of the computer programs developed, and on the other hand, the group members assessed their respective performance in the team. The project supervisors also took the quality of the reports into account for the overall assessment. An assessment of the learning benefits in the context of a comparative study, however, proves difficult. The student sample per project topic is too small, and there are no reference groups that do not participate in these projects. Nevertheless, the benefits of this learning format are evident, as our students are highly successful in international, competitive, and team-oriented formats in higher semesters, such as Formula Student [11] and Formula SAE [12].

Summary and Conclusions

In this paper, the numerical simulation of swarm behavior was presented as an example of the team-oriented project-based learning environment established at our institute. As part of this teaching and learning framework, our students work on application-related project tasks in the early semesters of their studies. These tasks are designed as multidisciplinary software projects that are part of the Computer Programming course and another course in the STEM field.

The project presented here is representative of about a dozen tasks that were worked on by our second semester students in the past academic year. The student teams were asked to simulate and visualize the behavior of a larger number of entities that interact only locally, following simple rules without central control. The simulations are based on the Vicsek and *boids* models and allow the user to vary all relevant model parameters as desired. Working on this project encouraged creativity and innovation. The students could experiment with different rules, parameters, and initial conditions, which was fostering a culture of exploration and innovation. Through their simulations, they could observe the emergence of collective phenomena that include cohesive motion, flocking, and self-organization. Thus, the project

provided insights into how simple local interactions can give rise to complex global behaviors, which has parallels in real-world systems.

The computer program created by our students offers a playful exploration of swarm behavior. It provides insights into the principles governing collective motion and might equip students with an appreciation for the elegance and complexity of natural phenomena. The program can be downloaded, free of charge, from the website https://fahrzeugtechnik.fh-joanneum.at/software/ExploringSwarmBehavior/, and can be executed on computers running the Windows operating system.

Authorship declaration and acknowledgments

The swarm behavior program presented in this paper was developed by Konrad Dobetsberger, Markus Ensbacher, and Alexander Strutzenberger under the supervision of Günter Bischof and Christian Steinmann. The last-mentioned authors (G.B. and C.S.) would like to express their sincere gratitude to all participating students of this year's undergraduate project-based learning program for their high motivation and excellent performance during their project work. The authors would also like to express their gratitude to Wolfgang Dautermann for arranging the website access to the program.

Bibliography

[1] Gabriel Popkin, *The physics of life*. Nature **529**, 16-18 (2016) DOI: 10.1038/529016a

[2] Marileen Dogterom and Stanislas Leibler, *Physical Aspects of the Growth and Regulation of Microtubule Structures*, Phys. Rev. Lett. **70**(9), 1347 (1993)

[3] François J. Nédélec, T. Surrey, A. C. Maggs, and S. Leibler, *Self-organization of microtubules and motors*, Nature **389**, 305–308 (1997).

[4] Tamás Vicsek, A. Czirók, E. Ben-Jacob, I. Cohen, and O. Shochet, *Novel type of phase transition in a system of self-driven particles*. Phys. Rev. Lett. **75**(6), 1226 (1995) DOI: 10.1103/PhysRevLett.75.1226

[5] Craig W. Reynolds, *Flocks, Herds, and Schools: A Distributed Behavioral Model*, ACM SIGGRAPH '87 Conference Proceedings, Computer Graphics **21**(4), 25–34 (1987) DOI: 10.1145/37401.37406

[6] Christopher G. Langton, *Artificial Life*. In The Proceedings of an Interdisciplinary Workshop on the Synthesis and Simulation of Living Systems, Addison-Wesley, 1-47 (1989)

[7] Emilia Bratschitsch, A. Casey, G. Bischof, and D. Rubeša, *3-phase multi subject project-based learning as a didactical method in automotive engineering studies*, Proceedings of the ASEE Annual Conference and Exposition, Honolulu, HI (2007)

[8] Günter Bischof, E. Bratschitsch, A. Casey, and D. Rubeša, *Facilitating engineering mathematics education by multidisciplinary projects*, Proceedings of the ASEE Annual Conference and Exposition, Honolulu, HI (2007)

[9] Francesco Ginelli, *The Physics of the Vicsek model*, Eur. Phys. J. Special Topics **225**, 2099–2117 (2016) DOI: 10.1140/epjst/e2016-60066-8

[10] V. Hunter Adams, *Boids algorithm - augmented for distributed consensus*, https://vanhunteradams.com/Pico/Animal_Movement/Boids-algorithm.html, accessed Jan 05, 2024.

[11] Formula Student, Institution of Mechanical Engineers, http://www.formulastudent.com/, accessed Jan 29, 2024.

[12] Formula SAE, SAE International, https://www.fsaeonline.com/, accessed Jan 29, 2024.