

Authoring Boids using RBF interpolation

DH2323 Computer Graphics and Interaction | Final project

David Giraldo | 950323-5336 | M's Interactive Media Technology

dgiraldo@kth.se | davidgiraldo@kth.se | <https://how-to-find-me.netlify.app/>

https://github.com/DavidGiraldoCode/p-authoring_boids_RBF_interpolation

KTH Royal Institute of Technology

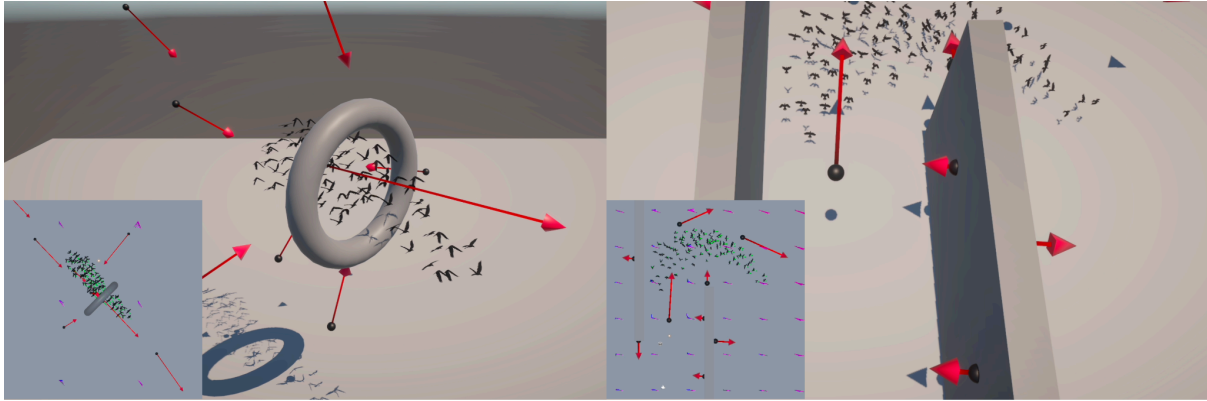


Figure 1: (Right) Boids flying across torus geometry influenced by a flow field using Gaussian RGB. (Left) Boids crossing two walls flying in an S shape are influenced by a flow field using Biharmonic RGB

Abstract

*Authoring the behavior of many virtual agents is time-consuming, involving multiple parameters and context-specific needs. Some steering algorithms use vector fields to influence agents' global paths. Jin's [JXJ*09] method stands out due to its use of Radial Basis Functions for gridless vector interpolation. This paper extends Jin's method to 3D vector fields for controlling the Boids algorithm by Reynolds (1998) and uses SteerBench test cases to evaluate this approach. Simulations showed Boids maneuvering through S shapes and shrinking to pass through narrow spaces. Implementation details and source code are available online.*

Authoring Steering Behaviors, Crowd Animation, Navigation Control, Radial Basis Functions, RBF, 3D Vector Field.

1. Introduction:

"Nature is the ultimate source of inspiration for computer graphics and animation" [Rey98], shedding light on what kind of virtual worlds and phenomena humans can create. Among some examples are movies like The Matrix Revolutions (2003) and The Lord of the Rings film series (2001 - 2003), in which designers have leveraged steering behavior algorithms to control multiple virtual agents on the screen.

The control of several virtual agents for scenario-specific scripts requires advanced technical knowledge and time-consuming parameter tuning [CFV*22]. However, multiple efforts have been made to make the authoring process more accessible to professionals unfamiliar with programming. For instance, sketch-like interfaces and drag-and-drop tools have emerged as effective solutions, enabling a more intuitive and accessible means of communicating creative intentions.

1.1. Motivation

This paper is driven by the possibilities that can be unlocked by exploring the authoring capabilities of

Radial Basis Functions and Vector Fields. Deriving from Jin's [JXJ*09] work on "Interactive Control of Large Crowd Navigation in Virtual Environments Using Vector Field" and Reynolds Steering Behavior [Rey02], this body of work delves into the intersection of Flow fields and the Boids algorithm, aiming at offering an accessible approach to authoring bird-like agents.

1.2. Research question

To what extent can RBF interpolation be used to author the global path of virtual agents that are not constrained to a 2D plane but move freely in 3D?

How can the resulting simulation using RBF be evaluated in terms of how close it can be to the author's creative intention?

2. Previous work:

Representing multiple living entities in a virtual world is used in a number of fields, from movies and video games to urban planning and architecture. And [LBC*22] Lemori's extensive categorization proves how committed computer graphics practitioners are to achieving plausible results. Rendering several virtual

agents is known as crow simulation, a branch of computer graphics animation, and it deals with representing non-verbal behaviors of virtual beings and their relations with their environment and others, characterized by the change of their position over time [CFV*22].

Raynolds' Boids model is a well-known steering behavior algorithm within the crow simulation field that simulates a flock of entities in a 3D digital environment. It has set a benchmark for what a user can do in terms of simulated animal behavior. He stated three main rules: collision avoidance, velocity matching, and flock centering, concepts that then were independently defined by Raynolds as Separation, Alignment, and Cohesion [Rey02]. There are several strategies to steer and author these boids that focus on the agent's local movement. However, as computer processing improves, game titles and movies strive for increasingly complex scenes where multiple agents interact and follow scripted behavior and paths.

Defining a virtual agent's behavior is known as authoring simulations, a multi-layer task that enables users to achieve creative intents and satisfy application-specific characteristics [LBC*22]. Six categories encompass the aspects that can be authored in a crow simulation: Hih-level behaviors, path-planning, local movements, body animation, visualization, and post-processing [LBC*22].

This paper focuses on Path-planing, which refers to authoring agents on a global scale, ideal when seeking to control large, endless crowds in complex environments not limited to a time window [LBC*22]. For global planning to happen, techniques such as Flows leverage vector fields that influence the path agents take without specifying individual paths explicitly. [LBC*22]

Vector field has been use to implement flow in the shape of 2D tiles on the ground [CH*04], potential fields using control points (Figure 2), sketch-based interfaces for defining flows, and derivations of vector field tiles that dynamically move with the agent [CFV*22]. All of them rely on a uniform grid to interpolate the vectors that influence agents' movement.

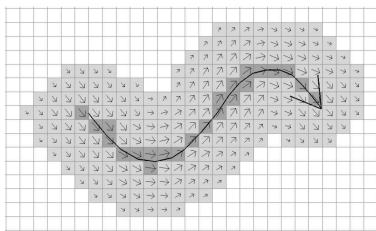


Figure 2: A user-defined flow on a uniform grid [PA*11]

[JXJ*09] presented a different approach that incorporated radial basis function (RBF) interpolation to create a grid-less vector field, which is then used to guide pedestrians' flow in a scene. The RBF approach allows users to input points (Figure 3), referred to as source points [EMB*17], that represent a vector at that location in space.

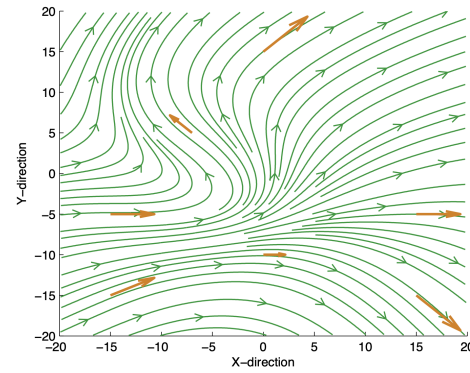


Figure 3: Vector field plot of a user-defined vector field with arrows showing the source points [JXJ*09]

His results demonstrated the authoring of crows with RFB vector fields as highly performant, fairly easy to implement, and reflexible enough to be combined with local moment algorithms like collision avoidance [Rey02]. Thus, the Boids algorithm is the perfect candidate for combining with RBF vector interpolation.

3. Radial Basis Functions (RFB) for vector field Interpolation in 2D and 3D

Having no sample grid is a scattered data problem for which traditional linear interpolation does not suffice. Thus, Radial Basis Functions exist as a solution for this problem by defining a function capable of interpolating any given discrete value in space, given all the values at source points [EMB*17]. The general equation (1) computes the Euclidean distance between a sample point (an agent's position on the scene) and each of the source points (defined by the user), multiplied by a weight λ .

$$s(x) = \sum_{i=1}^n \lambda_i \phi(||x - x_i||) \quad (1)$$

Since $s(x)$ uses the distance from a center point to all the other source points [SmSk16], it does not depend on a grid and can be implemented to interpolate using scattered 2D or 3D data, such as vectors (Figure 4).

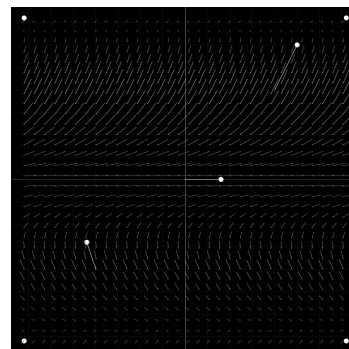


Figure 4: A vector field with four white source points with vector zero located on the corners and three white source points with vector different from zero.

The weights are fundamental for the interpolation as they can be understood as source points' "intensity" [EMB*17], determining the influence that every source point has on the agent's position. This means that not

only the distance between the sample and source points defines the result, as well as the vectors' magnitudes.

All weights λ_i are computed by solving the linear system of the equations (2), where Φ is a symmetrical coefficient matrix [Toi08], defining the relationship between all source points, also called distance matrix [DM18]. And f is the scalar value at a source point.

$$\Phi \lambda = f \quad (2)$$

Figure 5 clearly explains the explicit representation of the matrix, which can be solved using Gaussian Elimination.

$$\begin{bmatrix} \phi(\|x_1 - x_1\|) & \phi(\|x_1 - x_2\|) & \cdots & \phi(\|x_1 - x_n\|) \\ \phi(\|x_2 - x_1\|) & \phi(\|x_2 - x_2\|) & \cdots & \phi(\|x_2 - x_n\|) \\ \vdots & \vdots & \ddots & \vdots \\ \phi(\|x_n - x_1\|) & \phi(\|x_n - x_2\|) & \cdots & \phi(\|x_n - x_n\|) \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_n \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix}$$

Figure 5: Expansion of the coefficient matrix [Toi08]

Depending on the type of RBF, the resulting interpolation will yield different results in terms of the smoothness of the vector field. The Gaussian, Biharmonic, and Thin-Plate Spline (used by Jin's method) are examples of common types of RBFs.

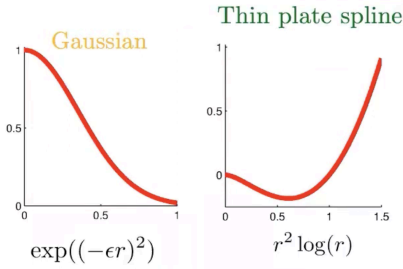


Figure 6: Smoothness and behavior of the RBF [Wri21].

Since f is a scalar value, the reader might have realized already that there is a step missing since the data at source points are vectors, not scalar values. Jin's paper explains that in order to interpolate a vector at a sample position, the algorithm needs three sets of RBFs, one for each vector's coordinate (x, y, z). Therefore, three equations are needed: (3), (4), and (5), where p is the position of the sample and p_i is each one of the source points.

$$s(p^x) = \sum_{i=1}^n \lambda_i^x \phi(\|p - p_i^s\|) = x \quad (3)$$

$$s(p^y) = \sum_{i=1}^n \lambda_i^y \phi(\|p - p_i^s\|) = y \quad (4)$$

$$s(p^z) = \sum_{i=1}^n \lambda_i^z \phi(\|p - p_i^s\|) = z \quad (5),$$

Which then allows the algorithm to use the resulting scalar value to instantiate a vector at the sample point (Figure 7).

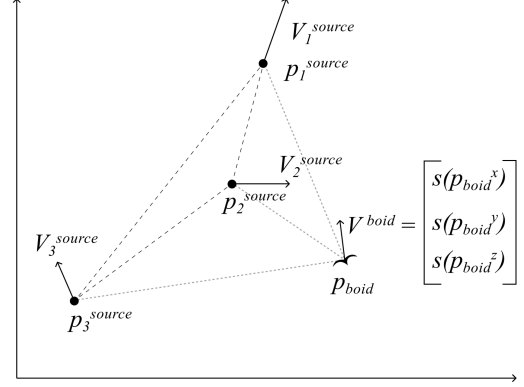


Figure 7: The three RBF interpolation functions for the scalar values x, y , and z form the vector at the Boid's position.

4. Implementation.

This study's implementation introduces an approach that combines Jin's flow field with Reynolds' Boids algorithm, aiming to create a more plausible and visually appealing result for bird-like agents. The system ensures that the Boids maintain a balanced distance from each other, avoiding both excessive spreading and collisions while following a predefined path. The simulation used C# in Unity 2022.3 LTS without the use of any additional package or third-party API. The source code can be found at the GitHub online repository 'p-bois_steering_behaviors'.

Jin's original equation (Figure 8) to interpolate the vector fields was altered to simplify the computation for the system of equations $\Phi f = \lambda$, enabling an ad hoc C# Gaussian Elimination solution. The coefficients matrix Φ , the RBF $\phi(\|x - x_i\|)$ (Code snippet 1), and the interpolations were also implemented in C# using Unity's scripting API.

$$\gamma(x) = P(x) + \sum_{i=1}^n \lambda_i (\|x - x_i\|_2)^2 \log(\|x - x_i\|_2)$$

$$\gamma(p^{xyz}) = \sum_{i=1}^n \lambda_i^{xyz} \phi(\|p - p_i^s\|)$$

Figure 8: Jin's original RBF equation (top) with a polynomial term for stability and the simplified version (bottom).

Code snippet 1: Computation of Phi using Unity and C#.

```
double Phi(Vector3 vector_j, Vector3 vector_i)
{
    Vector3 distance = vector_j - vector_i;
    float r = distance.magnitude;
    //Gaussian (GS)
    double GSKernel = Math.Exp(-0.001 * Math.Pow(r, 2));
    //Spline (S) biharmonic (φ(r) = r)
    double SKernel = r;
    //triharmonic (φ(r) = r³)
    double TSKernel = Math.Pow(r, 3);
    return GSKernel;
}
```

¹ https://github.com/DavidGiraldoCode/p-authoring_boids_RBF_interpolation

Authoring begins with the user defining the source vectors at source points using Prefabs in the Unity Editor. The prefab consisted of two GameObjects (tail and head), and its vector subtraction returns the vector at the source point (tail). A set of vectors is drawn on the Unity Editor viewport for debugging purposes; they do not affect the final result.

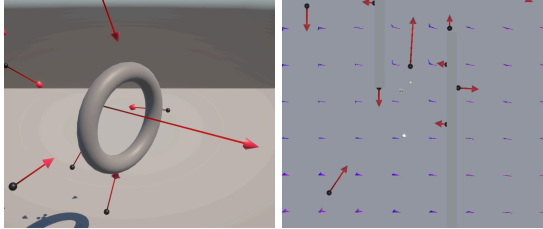


Figure 9: User-defined source vectors at source points to compute RBF interpolations at Boids' positions. The resulting vector field is visualized for debugging purposes.

Later, once the program starts, the first step is to compute the weights λ_i needed for interpolating sample points, as illustrated in Algorithm 1.

Algorithm 1 Computation of weights for RBF

Output: Source points and vectors

1. Store source points and vectors defined by the user
2. $X_{\text{Lamdas}} \leftarrow$ new array of sourcePoints size
3. $Y_{\text{Lamdas}} \leftarrow$ new array of sourcePoints size
4. $Z_{\text{Lamdas}} \leftarrow$ new array of sourcePoints size
5. Store coefficient Matrices Φ
6. $X_{\text{matrix}} \leftarrow \text{Compute}X\Phi(\text{sourcePoints}, \text{sourceVectors})$
7. $Y_{\text{matrix}} \leftarrow \text{Compute}Y\Phi(\text{sourcePoints}, \text{sourceVectors})$
8. $Z_{\text{matrix}} \leftarrow \text{Compute}Z\Phi(\text{sourcePoints}, \text{sourceVectors})$
9. **For each** coordinate x , y , and z , **do**
10. Compute Gaussian Elimination (XYZ_{matrix})
11. Solve for λ_{mdas} (XYZ_{matrix} , XYZ_{lamdas});
12. Store XYZ_{weights} in memory
13. **End for**

Output: Arrays of $XYZ_{\text{weights}} \lambda_i$

Once the weights λ_i are computed, the system can interpolate the vector at any position of the Boid, illustrated by Algorithm 2.

Algorithm 2 RBF Interpolation

Output: Position of a Boid

1. Store accumulators for X , Y , and Z interpolants
2. **For each** source point, **do**
3. $X_{\text{Interpolant}} += X_{\text{lamdas}}[i] * \Phi(\text{sample}, \text{source})$
4. $Y_{\text{Interpolant}} += Y_{\text{lamda}}[i] * \Phi(\text{sample}, \text{source})$
5. $Z_{\text{Interpolant}} += Z_{\text{lamda}}[i] * \Phi(\text{sample}, \text{source})$
6. **End for**
7. Instantiate the interpolatedVector using $\{X_{\text{Interpolant}}, Y_{\text{Interpolant}}, Z_{\text{Interpolant}}\}$
8. Return interpolatedVector

Output: The interpolated vector at that position

Lastly, the RBF interpolation is paired up with the Reynolds' Boids behavior in Algorithm 3 to add the flow field force to the Boid's movement.

Algorithm 3 Update Boids position

Output: Position of a Boid

1. **For each** simulation step, **do**
2. **For each** Boid in the flock, **do**
3. $\text{Acceleration} += \text{GetForceFromBounds}()$ *
4. $\text{Acceleration} += \text{GetConstraintSpeedForce}()$ *
5. $\text{Acceleration} += \text{GetSteeringForce}()$
6. $\text{Acceleration} += \text{GetForceFromVectorField}()$
7. Update Boid position $+= \text{Acceleration}$
8. **End for**
9. **End for**

Output: New position of the Boid

* $\text{GetForceFromBounds}()$ and $\text{GetConstraintSpeedForce}()$ are computed to restrain the Boids from flying away or increasing speed.

Birds' wings' flapping speed is controlled by the Boids' velocity, allowing them to flap their wings faster or slower according to the acceleration. ²Pulsar Bytes provided the bird mesh at the Unity Assets Store. Sound provided by ³SilentSeason at freesound.com.

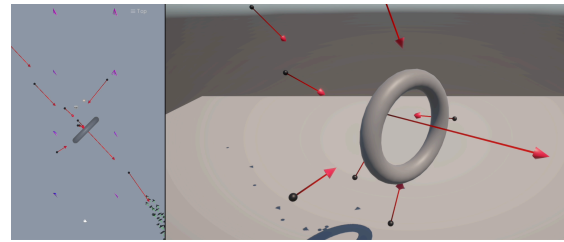


Figure 10: User-defined source vectors at source points to compute RBF interpolations at Boids' positions.

²<https://assetstore.unity.com/packages/3d/characters/animals/birds/low-poly-bird-raven-192706>

³ <https://freesound.org/people/silentseason/sounds/335871/>

5. Simulation results

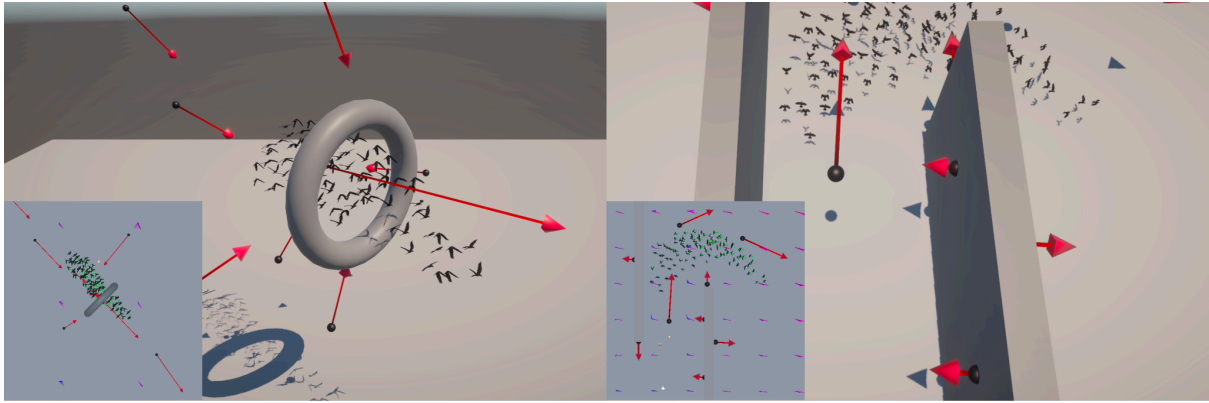


Figure 11: (Right) Boids flying across torus geometry influenced by a flow field using Gaussian RGB. (Left) Boids crossing two walls flying in an S shape are influenced by a flow field using Biharmonic RGB

Authoring Boids using RBFs proved to be a successful method, enabling detailed control of where, when, and how the flock of Boids should move. The first important finding for authoring Boids using this derivation of Jin's approach is that the source vector should not be seen as the path to follow but as forces that push the Boids. This means that depending on where the designer spawns the flock, the same vector field (combination of source vectors) will yield different results. Moreover, as mentioned in Jin's work, extending the RBF-interpolated flow field into 3D was effortless by solving for weights λy .

The Authoring capabilities of RBFs allow Boids to fly near each other to shrink the size of the flock and, therefore, cross narrow paths (Figure 11 - Right). This method also allows maneuverability in spaces with both shallow and steep turns (Figure 11 - Left) without collision. This is significantly important since there is no artificial field of view implemented on the Boids that allows them to detect the wall autonomously.

Upon further exploration, results showed that using a Gaussian RBF yields a more salient curve on the boid's overall flight (Figure 12). Due to its nature, the influence of the source points on the interpolated vector at the Boids's position decays with the distance, thus allowing the tree's main steering behaviors to contribute more to the simulation's next position.

An also interesting finding when applying the Gaussian kernel is that the forces keeping the boids inside the bounding box make them fly in circles once they reach the edges that define the flock flying area (Figure 13). While the biharmonic spline kernel pushes the boids outside the bounding area.

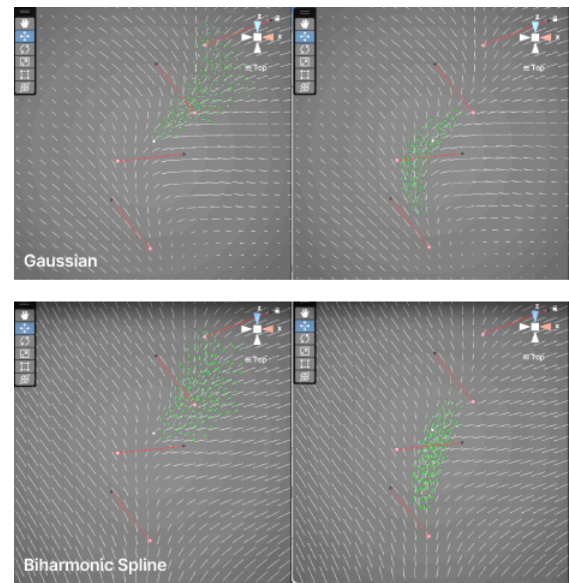


Figure 12: Differences in the resulting simulation when using Gaussian (top) and Biharmonic Spline (bottom) kernels.

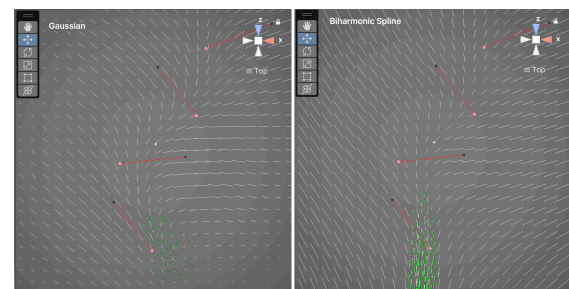


Figure 13: Influence of the bounding forces pending on the RBF's kernel, Gaussian (left) creates a vortex, and Biharmonic Spline (right) pushes the Boids outside the bounding area.

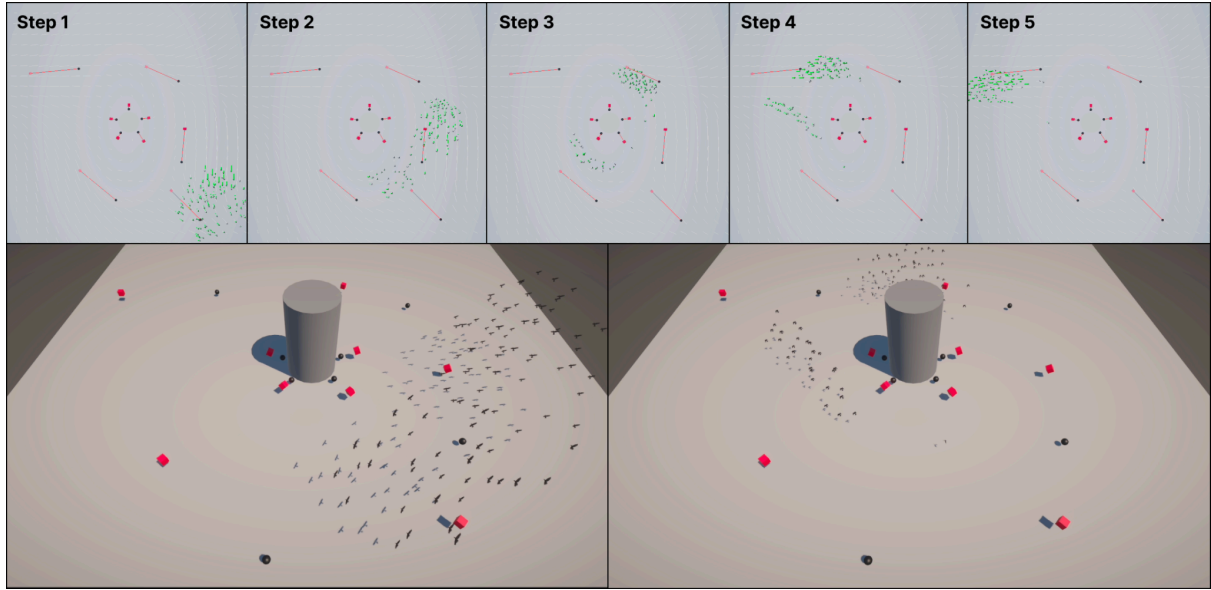


Figure 14: Flock of Boids crossing the scene and avoiding an obstacle (column) by following the vector field

A third simulation (Figure 14) demonstrated that even when artists strive for precise crowd control, the RBF method also affords flexibility, allowing Boids to separate into groups to take different routes and avoid obstacles.

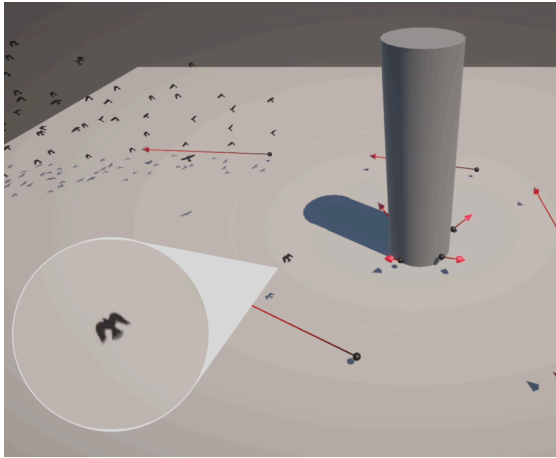


Figure 15: A flockmate is left behind and then rejoins the group.

Delving deeply into the frames, notice how one member of the flock is left behind after avoiding the obstacle, only to then reunite with the group. Considering that the original authoring intention did not plan for an abandoned Boid to happen, interesting behaviors like this raise questions about whether these types of incidents make a simulation more or less plausible.

6. Evaluation

6.1. Perceptual study:

After implementing the algorithm, the next logical step would be to evaluate the resulting simulation against well-known criteria. Steerbench [SS09], a widely recognized benchmark framework, could offer a starting point in this process. It computes several metrics to

evaluate how efficient an algorithm is in steering one or several virtual agents into completing a task related to a test case.

The framework's three main metrics are (1) Number of collisions (fewer, the better), (2) Time efficiency (the quicker to reach the goal, the better), and (3) Effort efficiency (the kinematic energy spent to reach the goal). The test cases are a set of scenarios consisting of what [SS09] Singh considers common situations that a digital agent might encounter and, consequently, an artist might need to reproduce by authoring the crowd's behavior.

Researchers who might want to conduct the aforementioned evaluation will soon realize that SteerBench might not be entirely suitable for testing the authoring capabilities of flow fields with RBF in the context of Boids. This is primarily because SteerBench gauges the efficiency of the steered agents over the visual outcome of the simulation. On the other hand, this paper focuses on how close the resulting steering behavior is to the designer's desired creative expression, disregarding efficiency since the most efficient path could be far from what the author is expecting. Secondly, Singh's framework aims to approximate humanoid agents that move on a 2D plane, while Reynolds' original Boids move on a 3D space. Ultimately, in [LBC*22] Lemonari's words, there is still the need for a method that can evaluate high-level aspects of steering behavior and not just efficiency.

Nonetheless, this paper has gathered a list of SteerBench scenarios (Table 1) that are suitable and could serve as a starting point to evaluate to what extent grid-less vector fields and RBFs can achieve different authoring results. [CFV*22] Colas tested Interaction Fields for authoring crowd simulation using a similar approach; they prompted users to author the behavior of a crowd based on a set of reference scenarios.


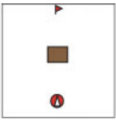
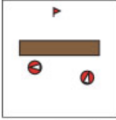
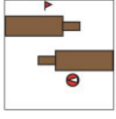
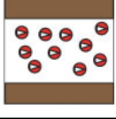
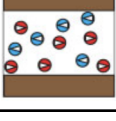
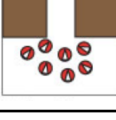
Scenario *Images from SteerBench	Description
	[#] Simple: The boid starts at the center, steering toward a target located to the left, right, or behind.
	[#] Simple obstacle: The boid starts on one side of the scene, steering towards a target located behind an obstacle on the other side of the scene.
	[#] Simple wall: Several boids steering around a wall to reach a target on the other side.
	[#] Curves: One or several boids steering through an S-curve to reach a target on the other side.
	[#] Hallway-one-way: Many boids flying in the same direction through a hallway delimited by two walls.
	[#] Hallway-two-way: Many boids flying in either direction through a hallway. Boid should form lanes.
	[#] Bottleneck-squeeze: All boids begin on one side of the scene and must enter and cross a hallway to reach the target.

Table 1: List of suitable scenarios to evaluate the authoring capabilities of flow fields using RBFs.

Therefore, the main objective of a possible experiment could be to **evaluate the authoring capabilities of flow fields using RDF to represent a steering behavior that approaches the artist's intended creative expression.** The experiment must gather at least five professionals in digital animation who are unfamiliar with programming to act as expert evaluators [HS10] of the authoring method.

As in [CFV*22] Colas user testing of IFs, each participant should undertake two phases. (1) Phase one will introduce them to how flow fields work on a general level and test what they understand. (2) Phase two will present the animator with a set of crowd-steering scenarios that they should recreate. For the test to run smoothly, it is appropriate that the scenarios, apart from being informed by Steerbench's common test cases, they are paired up with a video demonstrating the expected outcome, as shown in Figures 13, 14, and 15.

It must be stressed that the test disregards any usability issues since it is not evaluating the GUI but the resulting representation of a flock of birds in terms of how plausible it looks.

Test technical requirements	
Two screens	The first screen will display the tasks and expected outcomes of the simulation, while the second one will be for the user to execute the task.
PC with: <ul style="list-style-type: none"> - Unity installed. - A screen recording software or a program that can record the resulting simulation. - A mouse 	The editing of the source points and vectors will be carried out using Unity's editor, as well as the execution of the simulation.

Table 2: Minimum technical requirements to carry out the evaluation.

6.2. Discussion

At the present time, few, if any, developments have been published about authoring the 3D flow of Boids. Thus, it is important to recognize that the path-planning methods using 2D vector fields for controlling global paths may be limited in their ability to author the flow of bird or fish-like agents since they move using the three coordinates (X, Y, Z).

Flow fields only do not afford realistic behaviors; [CH*04] sometimes, they sacrifice a plausible result in return for a high level of control. [PA*11] Patil's work explains how Navigation Fields can be paired with traditional collision avoidance algorithms to produce a more realistic result. This is why the original Boids' three steering forces worked harmonically with Jin's approach since the RBF-interpolated vectors are added up on top of the separation, cohesion, and alignment forces.

Ultimately, Several factors account for plausible flock steering behaviors that can lead to further perceptual studies. From how steep the Boids can maneuver to the case in which one of them is left behind.

7. Conclusion and future work:

Based on the results this paper has yielded, there are several perspectives that computer graphics practitioners might be interested in pursuing, implementation-wise and evaluation-wise.

The first would be to afford complex math computations and exceptions. Unlike Jin's method, which relies on MATLAB API functions for RBF computation, this paper introduces an ad hoc solution in C# to solve the system of equations. However, it did not use a polynomial term, so the size of the coefficient matrix Φ was only $n*(n+1)$. Moreover, this algorithm presupposes that vectors are placed in a way that does not require handling mathematical exceptions. In all the previously presented simulations, all source points were no closer than five units between each other, and all source vectors were non-zero. If this technique were to be implemented in a commercial video game, the previous flaws would need to be addressed properly to avoid unwanted results.

Secondly, there is a lot that can be done in terms of defining a user-friendly interface to input and modify source vectors. For instance, industry crowd simulation

software, where authors need complete and precise control over the path agents follow, tends to use sketch-based interfaces where artists explicitly draw flows on the ground [LBC*22] or brush-like metaphors, as in CrowdBrush [UDT*04], to specify the type of input and its intensity. Simultaneously, other authors like [BABS20] Bönsch have proposed immersive VR experiences to enable authoring crowd paths.

Lastly, there is an opportunity to study how much crowd authoring evaluation, such as SteerBenchmark, can be extended to agents that move in 3D coordinates. Which ultimately increases the complexity of the test cases for which this framework should account.

In conclusion, this study proved that RBF is an effective method for authoring bird-like agents moving in a 3D, gridless space. After implementing a 3D-vector-field derived version of [JXJ*09] Jin's work, the algorithm was then integrated successfully with [Rey02] Reynolds' Boids algorithm to steer bird-like agents in different settings. Interestingly, different types of RBF yielded different results: Gaussian-RBF created more salient curves and vortex-like forces at the flock's boundaries, while Biharmonic-RBF created shallow turns but overpowered the flock's boundary forces. Moreover, a potential perceptual user study was proposed, gathering a set of suitable SteerBench test cases to evaluate RBF's authoring potential to create plausible results.

8. Acknowledgments

This project was supervised by KTH professor and course coordinator Christopher Peters.

References

- [BABS20] Bönsch A., Barton S.J., Ehret J., and Kuhlen T. W.. 2020. Immersive Sketching to Author Crowd Movements in Real-time. In Proceedings of the 20th ACM International Conference on Intelligent Virtual Agents (IVA '20). Association for Computing Machinery, New York, NY, USA, Article 11, 1–3. <https://doi.org/10.1145/3383652.3423883>
- [CH*04] Cheney S. 2004. Flow tiles. In Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation (SCA '04). Eurographics Association, Goslar, DEU, 233–242. <https://doi.org/10.1145/1028523.1028553>
- [CFV*22] Colas, A., van Toll, W., Zibrek, K., Hoyet, L., Olivier, A.-H., Pettré, J.: Interaction Fields: Intuitive Sketch-based Steering Behaviors for Crowd Simulation. *Computer Graphics Forum*, 41: 521–534. (2022). <https://doi.org/10.1111/cgf.14491>
- [DM18] De Marchi S.: Lectures on Radial Basis Functions. Department of Mathematics “Tullio Levi-Civita”, University of Padua, Italy, February 13, 2018.
- [EMB*17] Evangelos, M., Biancolini, M. E.: Fast Radial Basis Functions for Engineering Applications. Springer International Publishing AG, part of Springer Nature. (2017). https://doi.org/10.1007/978-3-319-75011-8_2
- [FFB*16] Flyer, N., Fornberg, B., Bayona, V., Barnett, G. A.: On the role of polynomials in RBF-FD approximations: I. Interpolation and accuracy. *Journal of Computational Physics*, 321, 21–38. (2016). ISSN 0021-9991. <https://doi.org/10.1016/j.jcp.2016.05.026>
- [HS10] Hwang, W., Salvendy, G.: Number of people required for usability evaluation: the 10±2 rule. *Commun. ACM* 53, 5 (May 2010), 130–133. <https://doi.org/10.1145/1735223.1735255>
- [JXJ*09] Jin, Xiaogang & Xu, Jiayi & Wang, Charlie & Huang, Shengsheng & Zhang, Jun. (2009). Interactive Control of Large-Crowd Navigation in Virtual Environments Using Vector Fields. *IEEE computer graphics and applications*. 28. 37–46. 10.1109/MCG.2008.117.
- [LBC*22] Lemonari, M., Blanco, R., Charalambous, P., Pelechano, N., Avraamides, M., Pettré, J., Chrysanthou, Y.: Authoring Virtual Crowds: A Survey. *Computer Graphics Forum*, 41: 677–701. (2022). <https://doi.org/10.1111/cgf.14506>
- [Nie94] Nielsen, J.: Estimating the number of subjects needed for a thinking aloud test. *International Journal of Human-Computer Studies*, 41(3), 385–397. (1994). ISSN 1071-5819. <https://doi.org/10.1006/ijhc.1994.1065>
- [PA*11] Patil, S. & van den Berg, Jur & Curtis, Sean & Lin, Ming & Manocha, Dinesh. (2011). Directing Crowd Simulations Using Navigation Fields. *IEEE transactions on visualization and computer graphics*. 17. 244–54. 10.1109/TVCG.2010.33.
- [PM*10] Park, M.J. Guiding flows for controlling crowds. *Vis Comput* 26, 1383–1391 (2010). <https://doi.org/10.1007/s00371-009-0415-4>
- [Rey98] Reynolds, C. W.: Flocks, herds, and schools: a distributed behavioral model. *Seminal graphics: pioneering efforts that shaped the field*, Volume 1. Association for Computing Machinery, New York, NY, USA, 273–282. (1998). <https://doi.org/10.1145/280811.281008>
- [Rey02] Reynolds, Craig. (2002). *Steering Behaviors For Autonomous Characters*.
- [SmSk16] Smolik M., Skala V.: Vector Field Interpolation with Radial Basis Functions. *The Swedish Chapter of Eurographics*, 2016.
- [SS09] Singh S., Kapadia M., Faloutsos P, and Reinman G. 2009. SteerBench: a benchmark suite for evaluating steering behaviors. *Comput. Animat. Virtual Worlds* 20, 5–6 (September 2009), 533–548
- [Toi08] Toit, W.: Radial Basis Function Interpolation [Master's thesis, University of Stellenbosch]. Applied Mathematics, Department of Mathematical Sciences, University of Stellenbosch. (2008).
- [UDT*04] Ulicny, B., de Heras Ciechowski, P., Thalmann, D.: Crowdbrush: interactive authoring of real-time crowd scenes. In Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation (SCA '04). Eurographics Association, Goslar, DEU, 243–252. (2004). <https://doi.org/10.1145/1028523.1028555>
- [Wri21] Wright, G.: Meshfree methods for scientific computing [Video]. YouTube. Meshfree Methods for Scientific Computing. (April 9, 2021).

Meshfree Methods for Scientific Computing