Department of Computer Science and Engineering

IIT Delhi

MATHEMATICAL MODEL OF A FLOCKING SIMULATION SOFTWARE

April 29, 2018

Divyanshu Saxena | Akshat Khare

Design Practices | COP 290

Contents

Introduction

We have to design and implement a software for Boid Flocking Simulation. Boids is an artificial life program, developed by Craig Reynolds in 1986, which simulates the flocking behaviour of birds. The software should have the following functionalities:

1. It should be able to approximate the flocking of starlings with some rules, that will form the model.
2. Given the model, we will computationally simulate the phenomenon by modeling each bird as an independent agent communicating and cooperating with other neighboring agents, adhering to the set of rules defined in our model.
3. It should be possible to measure/quantify the software simulation from a realistic flocking. This may make the use of average energy spend by each bird, the angular momentum and the force that each bird has to withstand in a typical flight ritual.

The remaining document contains the rules formulated for the model and other algorithmic optimisations.

FLOCKING RULES

The three most basic rules for simulation are as follows:

1. Alignment (d_{align})
2. Cohesion (d_{cohese})
3. Separation ($d_{separate}$)

The above rules define the motion of a single boid with reference to the entire flock. For every boid, the effect of other flock members can be formulated using the above rules and the parameters as mentioned in the corresponding parenthesis.
Each parameter denotes the distance, within which the boid-boid interaction shall be activev.
Typically, the order for the three distances is: $d_{separate} < d_{align} < d_{cohese}$.
Besides these, the following rules must also be implemented so as to bring about the simulated behaviour of the boids in presence of external factors:

1. Obstacle Avoidance
2. Boundary Restriction
3. Energy Considerations

Alignment

We will require these three set of informations for getting the information about the 3d object:

1. Set of labeled vertices as tuples of x, y and z coordinates.
2. Set of edges as pairs of labels of vertices connecting the extreme ends.
3. Set of planes as tuples of labels of vertices bounding that plane.

We will deliver the views as represented by three sets of information about the 2d representations:

1. Set of tuple of points labeled according to order of their increasing distance from the plane as tuples of x,y,z depending on the view of consideration.
2. Set of visible edges as pairs of vertices connecting the extreme ends, labeled in order of their increasing distance from the plane.
3. Set of hidden edges as pairs of vertices connecting the extreme ends, labeled in order of their increasing distance from the plane.

2D Projections to 3D Object Description

We will develop the 3d model based on two views(orthographic to each other) that are given by the user in two sets of information:

1. Set of points as tuples of x,y,z depending on the view of consideration. (Note: A single coordinate tuple may be labeled more than once according to order of the increasing distance from the projection plane.)

2. Set of edges as pairs of vertices connecting the extreme ends.

Each view shall require the above two sets to be specified to our mathematical model.

We will deliver the 3D object as represented by two sets of information about the 3d representations:

1. Set of labeled vertices as tuples of x, y and z coordinates.

2. Set of edges as pairs of labels of vertices connecting the extreme ends.

We plan to represent the data taken and given to user in the form of JSON objects or matrices in our software package because this kind of data representation is widely accepted and used. Presently, in our mathematical model, we'll use matrices to represent all sorts of informations unless otherwise specified.

Mathematical Model

Based on our Specifications, we develop two models to represent and formulate our descriptions from the given input. Here are the two mathematical models:

1. 3D Object Description to 2D Projection on a Plane

2. 2D Orthographic Projections to 3D Object Description

3D Object Description to 2D Projection on a Plane

Let us consider the input from the user as:

- The set $\mathscr{A} = \{A_i\}$ of vertices with labels as A_i each of which is represented as a column matrix containing three elements: $\begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix}$
- The set $\mathscr{E} = \{E_j\}$ of edges with each element representing an edge between two vertices as: $E_j = (A_i, A_k)$
- The set $\mathscr{F} = \{F_i\}$ of faces with each element as a tuple of vertices forming a face in the 3D object, such that $F_i = (A_{i_1}, A_{i_2}, \ldots, A_{i_k})$
- The projection plane on which the projection is to be taken, $\mathscr{P} = \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$

We begin the description of our projection by first, projecting each point A_i in $\mathscr{A}$ on the projection plane $\mathscr{P}$.
For this we use **Routine A** (mentioned later) to evaluate the projections of each of the points. We form the set $\mathscr{A}' = \{A'_i\}$, the set of the projected vertices on the plane $\mathscr{P}$.
Once, the set of projected vertices is found, we can correspond each edge E_j to E'_j, the corresponding projected edge, as: $E'_j = (A'_i, A'_k)$; thereby, constituting the set $\mathscr{E}' = \{E'_k\}$ of projected edges onto the plane of projection $\mathscr{P}$.
The above procedure gives us a raw two dimensional projection of the 3D object on our projection plane. Next, we aim to define the hidden points and edges in this projection. The procedure to determine hidden vertices is shown in formulated in **Routine B** (mentioned later). Later, we use **Routine C** to determine the hidden edges in our 2D projection.

Routine A

Let's consider a vertex $A_i = \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix}$ and a plane $\mathscr{P}$, whose equation is represented as: $\begin{bmatrix} a \\ b \\ c \end{bmatrix}^T \begin{bmatrix} x \\ y \\ z \end{bmatrix} = [d]$. Now, a general point on the perpendicular from A_i to $\mathscr{P}$ can be represented as: $\begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} + t \begin{bmatrix} a \\ b \\ c \end{bmatrix}$. The solution of this point with the plane $\mathscr{P}$ shall be the projection point of the vertex A_i, which can be evaluated using the equation:

$$\boxed{\begin{bmatrix} a \\ b \\ c \end{bmatrix}^T \left(\begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} + t \begin{bmatrix} a \\ b \\ c \end{bmatrix} \right) = [d]}$$

The solution to the above equation gives t which can be used to evaluate projected point A'_i for the vertex A_i.

Routine B: Procedure to determine hidden points in the projection of a 3D pbject on a projection plane

Let us take the projected point A'_i of the vertex A_i. Next, consider the set $\mathscr{F}.\forall F'_i \in \mathscr{F}'$ (the corresponding faces on the projected plane), we check if A'_i lies in the interior of F'_i, using the **Ray Casting Algorithm**, described in **Subroutine D** (mentioned later).
A_i shall be a hidden point iff it is shadowed by a face (including its boundaries), for which the projection A'_i must lie on or within the projected face F'_k and the points A_i and A'_i must be on opposite sides of plane containing the face F_k (Refer **Subroutine E** for details).
This determines whether the point A_i is hidden or not.

Routine C: Procedure to determine hidden edges in the projection of a 3D object

Consider the edge $E_j = (A_i, A_k)$ and the corresponding projection $E'_j = (A'_i, A'_k)$ on the plane $\mathscr{P}$. Next, we determine if the points A_i and A_k are hidden or not. If both the points are hidden, then E_j must definitely be hidden.
For other cases, we evaluate the following: $\forall F_j \in \mathscr{F}$, evaluate if the point A'_i is in the interior of the polygon described by F'_j. This can be evaluated using the formulation in **Routine B**.
If A'_i is in the exterior of the polygon described by F'_j, we move on to the next face $F'_{j+1} \in \mathscr{F}$, else we have the following cases:

- If the point A_i is behind the face F_j, we evaluate the point of intersection of the edge E'_j with the polygon described by the face F'_j in the 2D projection.
  We take the first point of intersection of the edge E'_j with the face F'_j. Let's call this point B'_i. The line segment A_iB_i shall be hidden, where B_i is the corresponding point of B'_i on the 3D object.
  To evaluate the remaining part B_iA_k, we follow Routine C for this dummy edge $E''_j = (B_i, A_k)$. We evaluate B_i using the following formulation:

$$B_i = \begin{bmatrix} x_{b_i} \\ y_{b_i} \\ z_{b_i} \end{bmatrix} = \frac{d_{B'_iA'_i}A_k + d_{B'_iA'_k}A_i}{d_{B'_iA'_i} + d_{B'_iA'_k}}$$

  where $B'_i = \begin{bmatrix} x'_{b_i} \\ y'_{b_i} \\ z'_{b_i} \end{bmatrix}$; $A_i = \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix}$; $A_k = \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix}$ and d_{XY} = distance between points X and Y.

- If, however, A_i lies on the face F_j then, we evaluate the relative position of the point A_k w.r.t the face F_j using the formulation established in **Routine B**. Following, we have two cases:
  - A_k is hidden behind the face F_j, in which case the edge $E_j = (A_i, A_k)$ shall be hidden behind the face F_j.
  - A_k is on or above the face F_j, in which case E_j cannot be hidden by the face F_j and we move to the next face $F_{j+1} \in \mathscr{F}$.
- If the point A_i is lies ahead of the face F_j, then we follow the above procedure with the point A_k.

Subroutine D: Determine if a point lies in the interior of a polygon

Consider the line $L_i : \begin{bmatrix} x'_i \\ y'_i \\ z'_i \end{bmatrix} + t \begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix}$; $(t \geqslant 0)$ from a point A'_i in a direction defined by the tuple (α, β, γ), where, $A'_i = \begin{bmatrix} x'_i \\ y'_i \\ z'_i \end{bmatrix}$ and $\begin{bmatrix} a \\ b \\ c \end{bmatrix}^T \begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix} = [0]$.

Next we consider the points of intersection of this line L_i with the polygon defined by F'_i. $\forall A'_l \in F'_i$, check if $A'_l = A'_i$, in which case the point A'_i is in the interior of F'_i, else consider two consecutive vertices A'_l and A'_m, which form an edge, say $E'_k = (A'_l, A'_m)$ on the face F'_i and check the relative sides of the two points w.r.t. the line L_i.

If the two points are on the same side, then we do not have a point of intersection of L_i with E'_k else we have one point of intersection if none of the points are on the line L_i and two

points of intersection (repeated root) if either of the points is on the line L_i.
Similarly, we evaluate all points of intersection of the line L_i with the polygon described by F'_i for $t \geqslant 0$.
If the number of points of intersection of this ray with our polygon is odd, then the point A'_i lies in the interior of the polygon defined by F'_i else it is on the exterior.

Subroutine E: Relative Position of a Point w.r.t. a Plane

Consider the plane $P : \begin{bmatrix} a \\ b \\ c \end{bmatrix}^T \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} d \end{bmatrix}$ and the points $A_i = \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix}$ and $A'_i = \begin{bmatrix} x'_i \\ y'_i \\ z'_i \end{bmatrix}$. The points A_i and A'_i are said to be on the same side of the plane P, iff

$$d_{A_i}.d_{A'_i} > 0$$

Otherwise, the points A_i and A'_i are said to be on the same side of the plane P, iff

$$d_{A_i}.d_{A'_i} < 0$$

where, $d_{A_i} = \dfrac{\left| \begin{bmatrix} a \\ b \\ c \end{bmatrix}^T \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} - \begin{bmatrix} d \end{bmatrix} \right|}{\left| \begin{bmatrix} a \\ b \\ c \end{bmatrix}^T \begin{bmatrix} a \\ b \\ c \end{bmatrix} \right|}$ and $d_{A'_i} = \dfrac{\left| \begin{bmatrix} a \\ b \\ c \end{bmatrix}^T \begin{bmatrix} x'_i \\ y'_i \\ z'_i \end{bmatrix} - \begin{bmatrix} d \end{bmatrix} \right|}{\left| \begin{bmatrix} a \\ b \\ c \end{bmatrix}^T \begin{bmatrix} a \\ b \\ c \end{bmatrix} \right|}$.

(Note: $|.|$ denotes the determinant of matrix and not the absolute value.)

This completes our model and description for the projection of a 3D object on a given projection plane.

2D Orthographic Projections to 3D Object Description

The orthographic views given by the user must be on the standard planes of reference. We describe our model, assuming that: 1. Top view is taken on $x - y$ plane. 2. Front view must be taken on $x - z$ plane.. Let us consider the input from the user as:

- The set $\mathscr{T} = \{T_i\}$ of lists each of which is a list of points projected on $x - y$ plane.T_i is therefore a list of points A_{T_j} each of which is represented as a column matrix containing three elements: $\begin{bmatrix} x_{Ti} \\ y_{Ti} \\ 0 \end{bmatrix}$. Each A_{T_j} corresponds to the projection of the point P_j on the top view,i.e., $x - y$ plane. The point in list must me in order of their distance from the plane. Canonically, $\mathscr{T} = \{[A_{T_1}, A_{T_2}, \ldots], \ldots\}$.
- The set $\mathscr{F} = \{F_i\}$ of lists each of which is a list of points projected on $x - z$ plane.F_i is therefore a list of points A_{F_j} each of which is represented as a column matrix containing three elements: $\begin{bmatrix} x_{Fi} \\ 0 \\ z_{Fi} \end{bmatrix}$. Each A_{F_j} corresponds to the projection of the point P_j on the top view,i.e., $x - z$ plane. The point in list must me in order of their distance from the plane. Canonically, $\mathscr{F} = \{[A_{F_1}, A_{F_2}, ..], ..\}$.
- The set $\mathscr{E}_{\mathscr{T}} = \{E_{Ti}\}$ of edges as seen in top view with each element representing an edge between two list of vertices as seen from top view as: $E_{Tj} = (T_i, T_k)$.
- The set $\mathscr{E}_{\mathscr{F}} = \{E_{Fi}\}$ of edges as seen in top view with each element representing an edge between two list of vertices as seen from top view as: $E_{Fj} = (F_i, F_k)$.

We begin the description of our project by first, coinciding the projections of the points from the tuples $\mathscr{T}_{\mathscr{T}}$ and $\mathscr{T}_{\mathscr{F}}$. For this we use **Routine F** (mentioned later). After going through this routine we are left with vertices of the 3d objects: $\mathscr{V}$. The process is not yet complete as we have not marked the edges till now. We will now use **Routine G**(mentioned later too). After going through this routine we have finally marked down the edges too. We are done describing the 3d object now.

Routine F: Determination of vertices

The underlying procedure makes use of the idea that if we draw line perpendicular to the planes of projections passing through the projected points they will coincide at points which correspond to the 3D description of these points. In essence, what is aimed is to evaluate the solution of the lines:

$$L_1 = \begin{bmatrix} x_{Ti_j} \\ y_{Ti_j} \\ z_{Ti_j} \end{bmatrix} + t \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} and L_2 = \begin{bmatrix} x_{Fi_j} \\ y_{Fi_j} \\ z_{Fi_j} \end{bmatrix} + t \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}.$$

This procedure evaluates the corresponding 3D point as: $\begin{bmatrix} x_{Ti_j} \\ y_{Ti_j} \\ z_{Fi_j} \end{bmatrix}$.

Note that: $x_{Ti_j} = x_{Fi_j}; y_{Fi_j} = 0; z_{Ti_j} = 0$.

Routine G: Determination of edges

We start off by determining the possible set of edges from a vertex, by taking the intersection of the connected neighbors (of a point $A_{T_j} \in \mathscr{T}$) in the top view and front view.
$\forall A_{T_j} \in \mathscr{T}$, determine the possible edges $E'_m = (A_{T_j}, A_{T_k})$ such that $E_m = (P_j, P_k)$ may be a possible edge in the 3D object. Similarly, we determine the possible edges $E''_m = (A_{F_j}, A_{F_l})$ To determine the sets $\{E'_m\}$ and $\{E''_m\}$, we follow the following procedure to determine these sets:

- $\forall A_{T_k} \in T_l$ other than A_{T_j} itself, $E'_m = (A_{T_j}, A_{T_k})$ must be included in the set of possible edges where $A_{T_j} \in T_l$.
- $\forall A_{F_k} \in F_l$ other than A_{F_j} itself, $E''_m = (A_{F_j}, A_{F_k})$ must be included in the set of possible edges where $A_{F_j} \in F_l$.
- $\forall A_{T_k} \in T_n$ such that there exists an edge $E'_m = (T_l, T_n) \in \mathscr{E}_{\mathscr{T}}$, E'_m must be included in set of possible edges. $A_{T_j} \in T_l$.
- $\forall A_{F_k} \in F_n$ such that there exists an edge $E''_m = (F_l, F_n) \in \mathscr{E}_{\mathscr{F}}$, E''_m must be included in set of possible edges. $A_{F_j} \in F_l$.

Now we have the set of possible edges, $\{E'_m\}$ and $\{E''_m\}$. Any edge in the 3D object must reflect in both of the two views in either of the ways mentioned above. The final set of possible edges, containing the point A_{T_j} must, therefore be $\{E'_m\} \cap \{E''_m\}$. This gives us the set of possible edges $\{E^j_p\}$ for the point P_j in the 3d object description. The next process would be to arrive at a set of definite edges $\{E^j\}$ from this set.
For this purpose we use three corollaries, mentioned below:

- *If $\exists T_i = [A_{T_j}, A_{T_k}] \in \mathscr{T}$, such that there is a possible connecting edge $E''_m = (F_l, F_n) \in \mathscr{E}_{\mathscr{F}}$ s.t. $A_{F_j} \in F_l, A_{F_k} \in F_n$ or vice-versa, then there must necessarily be an edge between the points P_j and P_k, the corresponding points in the 3D description. Similar argument holds for the tuple $\exists F_i = [A_{F_j}, A_{F_k}] \in \mathscr{F}$.* ***Corollary I***

- *If, in the set $\{E^j_p\}$, there exist more than one points such that, their 3D correspondences and the point P_j are collinear then P_j can only be connected to the next nearest point on*

this collinear line.
Hence, if the points other than P_j are on the same side of P_j, then an edge shall exist only between P_j and the nearest point.
However, if these other points are on different sides of P_j, then there can be an edge only between P_j and either of the two nearest points on each side. The other remaining points are discarded from the set $\{E_p^j\}$. ***Corollary II***

- *If $\exists$ points P_i and P_j, such that we have derived a definite edge between the two, and let there be a third point P_k such that $P_i \in \{E_p^k\}$ & $P_j \in \{E_p^k\}$, then there can be a definite edge between P_k and one of $\{P_i, P_j\}$. Note: The above two corollaries super-seed this one.* ***Corollary III***

We follow the above **Routine G** for every vertex P_i, and this routine is iterated until there is no change in any of the sets of definite edges $\{E_p^i\}\ for\ P_i \in \mathcal{V}$, the set of vertices in the 3D model description.
Hence we arrive at a simple wireframe model of the 3d object.
We delivered the set of vertices and set of edges which represent the 3d object.
This concludes our mathematical model of 3D to 2D as well as 2D to 3D reconstruction.

REFERENCES

1. **Ray Casting Algorithm**
   One simple way of finding whether the point is inside or outside a simple polygon is to test how many times a ray, starting from the point and going in any fixed direction, intersects the edges of the polygon. If the point is on the outside of the polygon the ray will intersect its edge an even number of times. If the point is on the inside of the polygon then it will intersect the edge an odd number of times.
   We used Point in polygon for learning about this algorithm.