Proceedings of the 2011 Winter Simulation Conference
S. Jain, R. R. Creasey, J. Himmelspach, K. P. White, and M. Fu, eds.

APPLICATIONS OF FLOCKING ALGORITHMS TO INPUT MODELING FOR AGENT MOVEMENT

Dashi Singham

Naval Postgraduate School
1411 Cunningham Road
Monterey, CA 93943, USA

Meredith Therkildsen

Naval Postgraduate School
1411 Cunningham Road
Monterey, CA 93943, USA

Lee Schruben

University of California at Berkeley
4141 Etcheverry Hall
Berkeley, CA 94720, USA

ABSTRACT

Simulation flocking has been introduced as a method for generating simulation input from multivariate dependent time series for sensitivity and risk analysis. It can be applied to data for which a parametric model is not readily available or imposes too many restrictions on the possible inputs. This method uses techniques from agent-based modeling to generate a flock of boids that follow the data. In this paper, we apply simulation flocking to a border crossing scenario to determine if waypoints simulated from flocking can be used to provide improved information on the number of hostiles successfully crossing the border. Analysis of the output reveals scenario limitations and potential areas of improvement in the patrol strategy.

1 INTRODUCTION

The idea of using flocking algorithms to generate simulation input was developed in Schruben and Singham (2010). Given a stream of real multivariate and dependent data, flocking algorithms can be used to generate new time series that appear to have similar statistical properties to the data. The method works by mapping each component of the n time series into a path in n-dimensional space, and simulating a flock of birds, called "boids", that follow the path of the data. These boid paths can be mapped back to the space of the data, and used as simulation input.

Input modeling is a major component of most simulation studies. Common options for generating input data include using real, or "trace", data, and fitting parametric models to data and then simulating from that parametric model. For military models, real data is often unavailable. But software exists to try to replicate real situations in real locations, often to a high degree of complexity. Many of these software programs use agent based modeling to simulate movement of forces in combat situations, but the user must decide ahead of time the agents' desired paths, marked by waypoints.

These waypoints are often hard-coded into the model if it is clear what the agents' objectives should be. For example, in a border crossing model, each red agent wishes to advance undetected across the border, while the blue agents patrol back and forth along the border. The locations of the waypoints can have a great influence on whether or not the mission succeeds. Models can include search agents that detect the agents trying to cross. Complex terrain may also be included in the model if data is available for the geographic region being studied. The presence of mountains or other natural obstacles can slow down or impede the movement of agents. The locations of the waypoints with respect to these obstacles can greatly

influence the probability of a successful crossing. Fixing the location of the waypoints only allows us to see the results when agents move along that specific path. If we are interested in designing a patrol system to prevent crossing, we should be interested in all possible movements the red agents could make.

One way to simulate possible waypoint paths in order to test the capabilities of the patrol strategy is by using simulation flocking. We start with a base case for the waypoints of the red agents: each agent has a straightforward path to their goal across the border. We use ideas from flocking, introduced by Reynolds (1987), to generate new waypoint paths (called "boid paths" in this paper) that follow the base case data to different degrees of closeness, while attempting to maintain the dependence and properties of the base case paths. The paths generated will still often move towards the target, but they may not be as direct as the base case path. We also allow the red agents to reverse their direction. We run our program using these simulated paths to see how well the model performs. Simulation flocking has been proposed as a method for generating input from trace data for sensitivity analysis. By applying different possible paths to our model, we can see how well our patrol strategy works across different scenarios.

Our flocking method provided an easy way of generating multiple paths to simulate the potential movement of red agents across the border. We used these simulated paths to test the capabilities of the model. Running the model with the simulated data revealed which types of paths were more likely to lead to agent success. We were also able to use flocking to reveal many problems with the model. By varying the flocking parameters, we could control how far the agents strayed from the conventional paths. Some of these paths resulted in configurations that we had not anticipated, leading us to re-evaluate the model and the patrol strategy.

2 MODEL DESCRIPTION

MANA (Map Aware Non-uniform Automata) is an agent-based, time-stepped, distillation model developed by the New Zealand Defence Technology Agency (DTA) for the New Zealand Defence Force. The model was built on the idea that overly detailed models are not helpful in finding robust system settings for desired battlefield outcomes, because they are too focused on extraneous issues (Lauren and Stephen 2002). MANA, therefore, models only the essential details of a scenario and tries to create a complex adaptive system that mimics real-world factors of combat. The agents are "map aware" meaning that the map serves as the agent's impression of its environment (McIntosh, Galligan, Anderson, and Lauren 2007).

For our experiment, MANA was used to build the simulation for applying the flocking algorithms. The scenario deals with border crossing along the Chile and Peru border. The map consists of the border between the two countries and includes geographical features, such as mountains and water. There is one major road running from north to south. The red agents start in Peru, and attempt to move south into Chile. There are ten squads in the scenario, including:

- One blue foot patrol squad of five agents who patrol the border along both sides at 3.75 miles/hr searching for red agents trying to illegally cross the border from Peru into Chile.
- One blue Quick Reaction Team (QRT) squad of two agents who patrol the border along both sides at 25 miles/hr also searching for red agents trying to illegally cross the border from Peru into Chile.
- One red squad of seven hostile agents who try to cross the border at 3.75 miles/hr without being classified by the blue force.
- Seven dummy agent squads each consisting of one agent who is stationary throughout the entire scenario and serves as a specific target for each of the red agents to shoot if they get close to their respective targets.

All the agents in the blue foot patrol squad, the blue Quick Reaction Team squad, and the red squad carry weapons and are able to shoot at each other. If a red agent successfully crosses the border, then it shoots its corresponding dummy agent at its final waypoint to signify its success. If a foot patrol or QRT vehicle classifies a red agent, then it shoots the red agent, signifying apprehension and no border

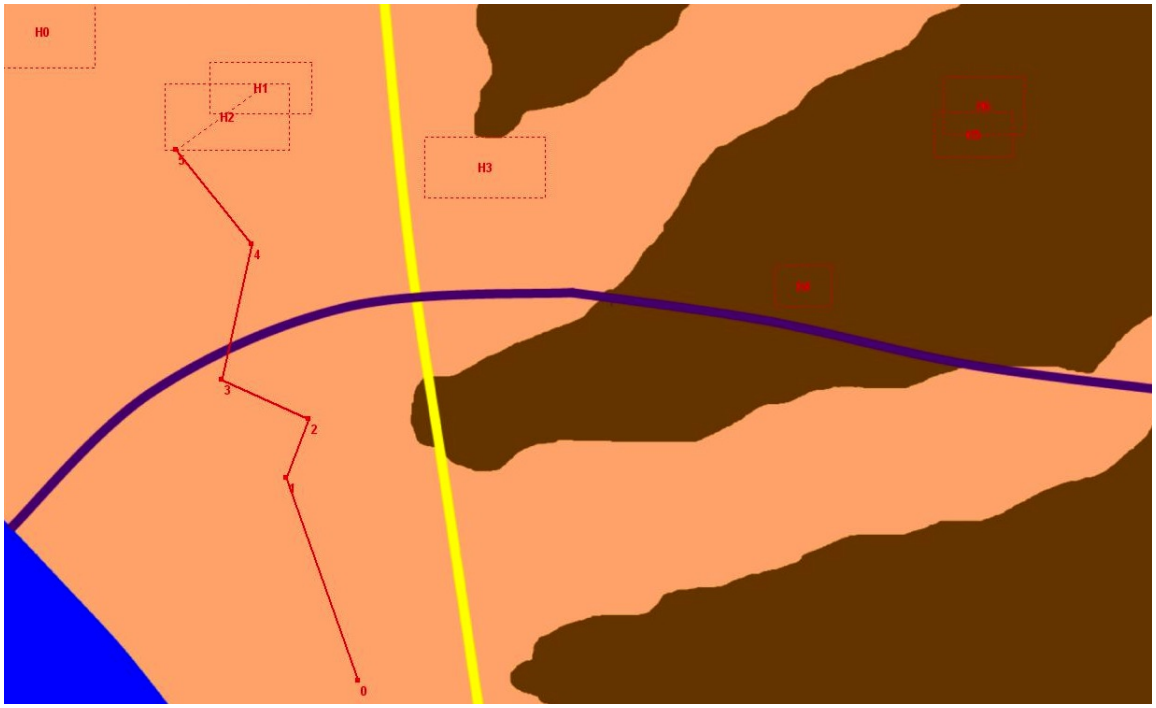

Figure 1: Plot of the border (east-west line) with a sample waypoint path for a red agent who is attempting to cross south into Chile. The boxes show various starting locations of the red agents.

crossing for that red agent. The final waypoint is stationed well beyond the border, so agents who succeed in crossing do not necessarily make it to their final target.

The randomization built into MANA occurs in the initial placement of agents in the blue foot patrol squad, the blue Quick Reaction Team squad, and the red hostile squad. The flocking algorithm parameterizes the waypoints for the red agents, thereby varying their paths across the border, to see if border crossing success is affected for each of the red hostile agents. Figure 1 shows the map used with the border extending from left to right, with Peru to the north and Chile to the south. The darker regions correspond to mountains, which slow the progress that can be made by the red agents on foot. The red path shows a possible set of waypoints of a red agent attempting to move south into Chile. The map covers an area of 12.17 miles by 8.21 miles, and the model simulates two to three hours of real time and takes approximately one minute to run.

3 FLOCKING ALGORITHMS FOR INPUT MODELING

In this section, we describe the flocking algorithm used to generate the waypoint data and to test the simulation model. It is an improved version of the algorithm in Schruben and Singham (2010). Our original data consists of a choice of paths across the border based on the authors' experience with other military models. First we start by rescaling the terrain map coordinates to the unit square. This allows us to use general flocking algorithms that do not depend on the scale of the map. We have seven red agents, each with a path given by waypoints. These waypoints consist of x and y coordinates. We have fourteen total time series: one for each agent's x-coordinates of their waypoints, and one for each agent's y-coordinates.

Next we map the rescaled time series into a path in the space $[0,1]^{14}$. This involves taking the fourteen points associated with each time step in the waypoint series and using those rescaled values as coordinates for points in the path in $[0,1]^{14}$. Each agent has six waypoints, plus a starting point, leading to seven points in the series. The result of the mapping is a seven point path in $[0,1]^{14}$, which we call the leading boid.

We apply the following flocking algorithm to this leading boid to generate followers, who will then be mapped back to real data (x and y-coordinates of waypoints).

The flocking algorithm has two parameters that model behavioral structure and randomness. The first parameter, λ, represents the affinity of the simulated boid to the leading boid. A high affinity means that the simulated boid follows the leading boid closely, whereas a low affinity means that the boid travels in a different (random or opposite) direction from the leading boid. If $\lambda = 1$, the simulated boid travels towards the exact location of the next point in the leading boid path. If $\lambda = -1$, it travels in the exact opposite direction, and if $\lambda = 0$, it travels in a completely random direction. Here, we take λ to be the proportion of the direction that is determined by location of the leading boid. If λ is negative, we invert the sign and apply that proportion in the opposite direction of the leading boid.

The second parameter in the flocking algorithm is a measure of the variance of the additive noise along the x and y dimensions. Let the noise term be independent and normally distributed with mean zero and variance σ^2. Then the second parameter of this algorithm is σ, since higher values of σ allow for more variation in the path of the simulated boids. By modifying the parameters λ and σ we can control how closely the simulated boid follows the leading boid. If $\lambda = 1$ and $\sigma = 0$, then the simulated boid path is exactly that of the leading boid. Generally, for high values of λ and low values of σ, the simulated boids will follow the leading boid closely.

Let $\mathbf{v}_t$ and $\mathbf{v}_{t+1}$ be fourteen dimensional vectors of the base case rescaled waypoint coordinates of the red agents for waypoints t and $t+1$, where $t \in \{1,2,\ldots,6\}$. Suppose we have simulated the vectors of the coordinates for a boid path for waypoints $1,\ldots,t$ called $\mathbf{w}_1,\ldots,\mathbf{w}_t$. We wish to generate the vector of the coordinates for the next waypoint, $\mathbf{w}_{t+1}$. Let $\theta_{t,t+1}$ be the normalized vector from $\mathbf{w}_t$ to $\mathbf{v}_{t+1}$ in Euclidean space, and let $r_{t,t+1}$ be the Euclidean distance between the points $\mathbf{w}_t$ and $\mathbf{v}_{t+1}$. Finally, let $\mathbf{z}_{t+1}$ be a vector of fourteen i.i.d. standard normal random variables, and let $\phi_{t,t+1}$ be a randomly generated normalized vector in $[0,1]^{14}$. Then, we can generate the next waypoint $\mathbf{w}_{t+1}$ according to the following equation:

$$\mathbf{w}_{t+1} = \mathbf{w}_t + r_{t,t+1}\left[\lambda\theta_{t,t+1} + (1-\lambda)\phi_{t,t+1}\right] + \sigma\mathbf{z}_{t+1}. \tag{1}$$

The algorithm initializes the location for each agent by randomly perturbing the base case starting points by a normal random variable with variance σ^2 in each of the x and y directions. Then, (1) is used to sequentially generate the next waypoints. The direction of the next waypoint is based on a linear combination of the direction to the next base case waypoint, and a random direction $\phi_{t,t+1}$. If λ is higher, then more of the direction is based on the base case data. This directional vector is scaled up by $r_{t,t+1}$ to match the scale of the distance between waypoints. Finally, an additive noise of $\sigma\mathbf{z}_{t+1}$ is used to generate additional randomness in the paths, with the scale of the deviations determined by σ. The flocking algorithm accounts for the dependence between agents by treating the multivariate data as a single path, so the simulation of one agent's location depends on the locations of the other agents.

We can vary σ and λ to generate paths that vary from the original leading boid path derived from the base case waypoints. Paths that follow the leading boid closely can be used to assess the sensitivity analysis of the model to the paths chosen by model experts. We generate these paths using high values of λ (to follow the leading boid closely) and small values of σ (to produce small random perturbations in the path). Varying the waypoints slightly may or may not lead to substantially different results. We can also use low values of λ, and high values of σ, in order to generate paths that appear to go in totally different directions. This allows us to see how the model will react for different scenarios, and see if there are alternative paths available that we are not considering in the base case scenario. In reality, the blue agents may not be able to control or anticipate the movement of the red agents, so it is important to see how robust our model is to different scenarios.

4 RESULTS

We used the flocking algorithm described above to generate waypoint paths for the red agents to use in our MANA model. We tried different parameter choices in order to obtain a range of possible paths for the

agents. The parameter choices are summarized in Table 1. For high values of λ, the simulated boid paths follow the lead boid closely. For lower values of λ, the paths may move in even the opposite direction of the leading boid (which is chosen by the model experts). For higher values of σ, the additive error to the paths has a higher variance, so paths can appear more jagged. For each parameter combination in Table 1, we generated 40 boids, and for each of those boids, we simulated 30 replications in MANA. MANA inserts its own randomness by using different seeds to change the initial locations of the agents.

Table 1: Parameter choices applied to the model with overall results.

Group	λ	σ	Deviation	Final Waypoint Across Border	Agents Crossing Successfully
1	0.9	0.02	Least Extreme	100%	49%
2	0.9	0.09	Intermediate	99%	21%
3	0.5	0.02	Intermediate	80%	27%
4	0.5	0.09	Most Extreme	68%	11%

Group 1 boid paths follow the base case waypoint most closely, with 90% of the direction determined by the base case data and a low value of σ for the additive error. Group 4 contains the boids that stray farthest from the base case path with lower λ and higher σ. We found these parameters were able to push the limits of what actual waypoints could be without leading to overly extreme results (like all the agents immediately traveling off the map). For each group, we show the percentage of paths that have a final waypoint across the border. Some of the boid paths generated were so extreme that the agent did not end up on the correct (south) side of the border. We also give the percentage of agents who crossed the border successfully. In order to have a successful crossing, the agents must reach their final waypoint, and their final waypoint must be across the border.

We found that the Group 1 boid paths result in the highest percentage of agents succeeding in crossing the border, and in reaching their final waypoint, while the Group 4 boid paths resulted in the lowest percentage of successes in terms of border crossing and reaching the final waypoint. Since the Group 1 boid paths are more direct, the agents are able to cross and reach their goal more often, whereas the Group 4 boids often end up taking longer indirect paths to their target, increasing their risk of capture and increasing their chances of not reaching their final waypoint before the scenario maximum time length was reached. For all groups, it appeared that while a high number of agents' paths led them across the border, a much smaller number was able to progress far enough past the border to reach their final waypoint.

Running the MANA model with the flocked waypoints revealed two different types of problems. The first type of problem was with the way our model was constructed. We found that while our model could handle the typical waypoint path of the leading boid for the base case, waypoints generated using $\lambda = 0.5$ and $\sigma = 0.9$ (Group 4) resulted in agents moving in unpredictable ways. They might cross the border successfully and then return back to the Peru side, making it hard to determine at the end of the experiment if they had successfully crossed. They might also end up traveling into the mountains, which slowed down their travel time resulting in incomplete runs due to agents not being able to move towards all their waypoints in time. The model would also fail to complete because the agents were moving in the opposite direction of the border, or their paths were so extreme and jagged that it took a long time before they reached their next waypoint.

As an example, Figure 2 shows a red agent that travels the wrong way into the mountains after almost crossing the border, and never makes it to the final waypoint even though their projected boid path sends them across, because the model runs out of time. Additionally, the path of the pictured red agent actually comes close to the dummy agent for another red hostile agent. The agent in this plot ends up killing that dummy agent making it appear that another agent successfully crossed the border. This revealed another issue with our model, in that it was hard to correctly determine which agent crossed the border due to the paths crossing.

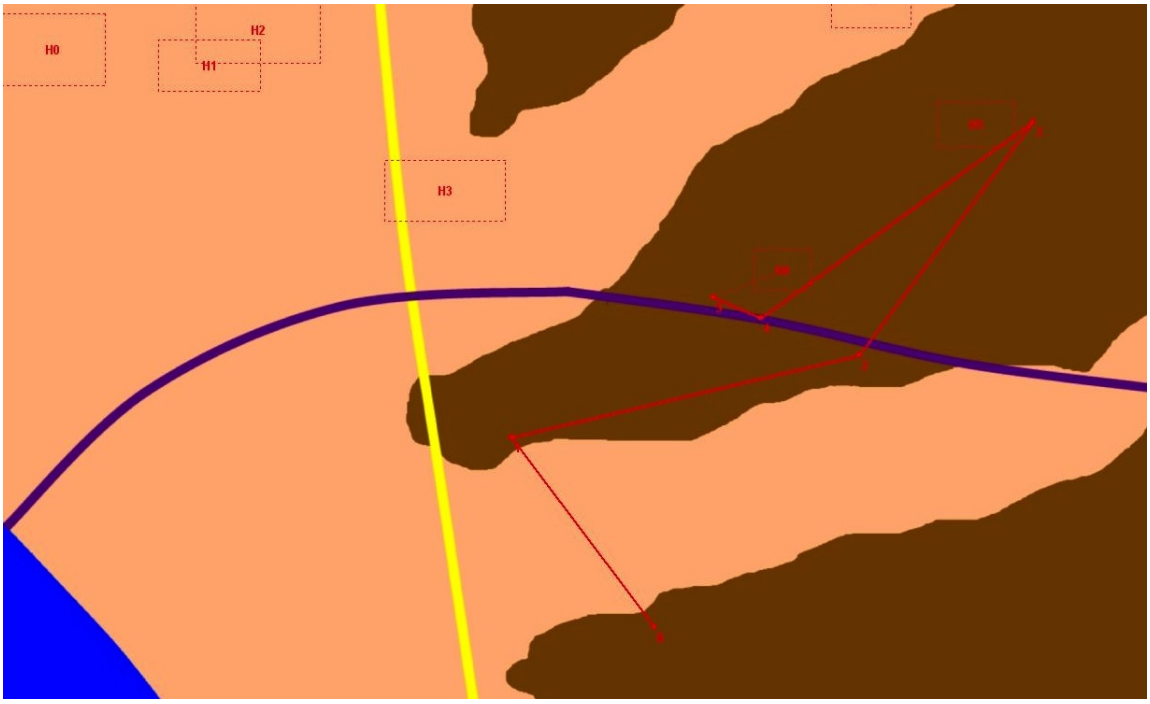

Figure 2: Plot of the path of a boid generated by the Group 4 flocking algorithm. The agent travels in the wrong direction, hindering our ability to track whether it crosses successfully and the agent's ability to complete its path in the maximum number of time steps for the scenario.

The second type of problem related to the strategy employed by the patrol. For all groups of boid paths, the red agents that started at the edges of the map (at the left towards the water or at the right towards the mountains) were the most successful in crossing and reaching their target undetected. This tells us that while the patrol strategy is doing an adequate job apprehending the red agents traveling in the center of the map, they are not spending enough time at the edges. This was true even for the Group 4 boid paths, where the boids did not necessarily travel directly along the edges of the map. Since we varied the boid paths to extreme levels in Group 4, we feel comfortable in concluding that the patrol strategy should be improved at the edges of the map, and that this result is not sensitive to the initial base case choice of waypoints of the red agents.

5 SUMMARY

Flocking proved to be a feasible way of simulating possible waypoint paths for the red agents in a border crossing scenario. By varying the flocking parameters, we were able to generate scenarios that were unlikely to happen, but revealed problems in the way our model was accounting for successful crossings. Randomness could be incorporated along the entire paths of agents, instead of just at their initial starting points. We were able to see how the patrol strategy performed for a variety of possible scenarios, and determine points of weakness in the blue strategy. Simulation flocking holds promise for military modeling since parametric models are often not available to model input data, and flocking can be used to test the robustness of models to different scenarios.

ACKNOWLEDGMENTS

We are grateful to Mary McDonald of the SEED Center at the Naval Postgraduate School for her help and advice in designing this model.

REFERENCES

Lauren, M., and R. Stephen. 2002. "Map-aware Non-uniform Automata (MANA)-A New Zealand Approach to Scenario Modelling". *Journal of Battlefield Technology* 5:27–31.

McIntosh, G., D. Galligan, M. Anderson, and M. Lauren. 2007. "MANA (Map Aware Non-Uniform Automata) Version 4 User Manual". *DTA Technical Note* 3:1–7.

Reynolds, C. 1987. "Flocks, herds and schools: A distributed behavioral model". In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, edited by SIGGRAPH '87, 25–34. ACM.

Schruben, L., and D. Singham. 2010, December. "Simulating Multivariate Time Series Using Flocking". In *Proceedings of the 2010 Winter Simulation Conference*, edited by B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E. Yücesan, 1048–1054. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

AUTHOR BIOGRAPHIES

DASHI SINGHAM is a Research Assistant Professor in the Operations Research Department at the Naval Postgraduate School and is working with the Simulation Experiments and Efficient Design (SEED) Center for Data Farming. She holds a Ph.D. in Industrial Engineering & Operations Research and an M.A. in Statistics from the University of California, Berkeley. She received a B.S.E. in Operations Research & Financial Engineering from Princeton University. Her research areas of interest are simulation input and output analysis, applied probability and statistics, and statistical computing. Her email address is dsingham@nps.edu.

MEREDITH THERKILDSEN is a Research Assistant at the Simulation Experiments and Efficient Design (SEED) Center within the Operations Research Department at the Naval Postgraduate School (NPS). She holds a B.S. in Mathematics from Loyola University Chicago. Her main research areas at NPS are agent-based combat modeling and human social culture behavior (HSCB) modeling. She mainly works with the Map Aware Non-Uniform Automata (MANA) model and the Peace Support Operations Model (PSOM). Her e-mail is matherki@nps.edu.

LEE SCHRUBEN is a Chancellor's Professor and past Chairman in the Department of Industrial Engineering and Operations Research at the University of California at Berkeley. Prior to joining the Berkeley faculty, he was on the Operations Research and Industrial Engineering faculty at Cornell where he held the Schultz Endowed Professorship in Engineering. He received his Ph.D. from Yale and is a Fellow of the Institute for Operations Research and Management Science. Professor Schruben's research interests are in simulation modeling and analysis methodologies with a broad range of applications, most recently focusing on biopharmaceutical production and supply chains. His email address is lees@berkeley.edu.