# Pattern Formation: Investigating Boids Algorithm for Multi-Robot Systems

Sahil Patel
*Arizona State University*
smpate11@asu.edu

## I. Abstract

Taking inspiration from Martian rovers, the main objective of the project is to study the pattern formation behavior of a group of robots that traverses through a confined space containing numerous obstacles. Utilizing Craig Reynolds' Boids algorithm, this study aims to understand how existing and self-developed mechanisms control the collective behavior within this multi-robot system. The mathematical model developed defines a wide range of parameters to satisfy multiple project objectives such as: collision avoidance, border avoidance, speed limiting, and velocity alignment.

The significance of investigating these behaviors can be applied in various fields including but not limited to: search and rescue operations, police surveillance activities, exploration assignments, and mapping missions. After analyzing and simulating this model, it was shown that the multiple groups created within the system converged to a local state of equilibrium and successfully incorporated a new rule that allowed the robots to change their paths whenever they detected an obstacle near them.

By studying this model's ability to create a cohesive behavior within a group, this project contributes to adding insights that might be beneficial for enhancing the efficiency and adaptability of these robotic swarms in real-world scenarios. This contributes to a more comprehensive understanding of the dynamics of these multi-robot systems in complex environments.

## II. Mathematical Model

This report investigates the pattern formation behavior for this multi-robot system. More specifically, by using a well-known model created by Craig Reynolds, called the Boids algorithm, this project will simulate the collective behaviors for this system. The objective is to understand how a group of robots can be able to organize themselves and coordinate their movements in order to achieve a defined flocking behavior while they traverse their environment.

### A. Hypothetical Scenario

To apply the pattern formation behavior, facilitated by the use of the Boids algorithm, a Mars exploration mission scenario was created. For this mission, a fleet of exploration rovers are deployed to explore and navigate a selected area on the Martian planet. During their mission, the rovers are tasked to collect valuable data while avoiding the area's obstacles. Whenever they are in a group, they are to collaboratively distance themselves in order to avoid colliding into each other, align their velocities with the average group velocity, and maintain unity.

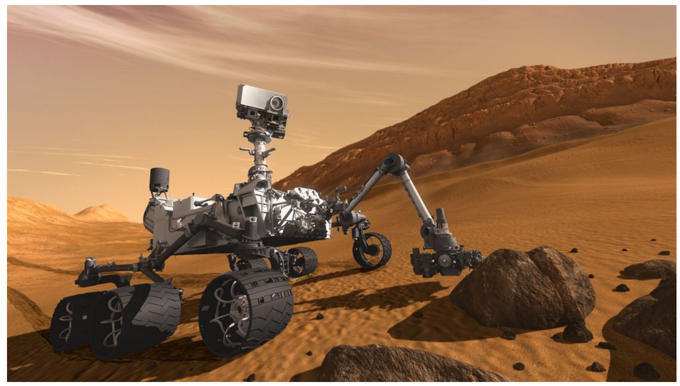

Fig. 1: An artist's rendition of the Mars Curiosity Rover [9].

### B. Assumptions and Constraints

The dimensions of the environment are defined to occupy a 100-meter by 100-meter area for the mission. This bounded space serves as the exploration area, and procedures have been implemented to ensure that the rovers do not accidentally venture beyond this region. Therefore, the rovers are programmed to adjust their trajectories when they approach the area's borders to prevent this situation from occurring.

Within this environment, the rovers are in constant communication with a ground station that is situated outside the exploration area. This communication channel is established primarily for exchanging information. This enables the ground station to relay data about the locations of obstacles (that are randomly scattered throughout the area) and positions of other rovers (also randomly scattered). It is crucial to note that this communication does not serve as a controller; instead, its purpose only enhances the situational awareness and decision-making for the rovers.

Scattered throughout the exploration area are obstacles in the form of rocks and boulders. These obstacles are the same size and distributed in a manner that pose as navigational challenges for the rovers, a fundamental aspect of the mission's objectives.

The rovers deployed for this mission are equipped with a set of instruments and sensors designed to enhance their capabilities. These include tools such as the Mastcam (Mast Camera), MAHLI (Mars Hand Lens Imager), GPS (Global Positioning System), odometers, accelerometers, gyroscopes, and ultrasonic sensors [8]. These instruments are necessary for the rovers to be able to capture valuable data, record their trajectories, avoid colliding with obstacles and other rovers, maintain unity, match speed, and gather essential information as they traverse the environment.

### *C. Model*

Fig. 2: A flock of birds [12].

Developed by Craig Reynolds in 1986, the Boids algorithm is a multi-robot model that simulates the flocking behavior of birds, fish, and other flocking animals. It uses a set of rules to guide the individual robots (or "boids") in a technique that results in emergent flocking behavior [3]. These core rules of the algorithm involve separation, alignment, and cohesion, all of which are discussed later in this report.

The mathematical equations presented in this section are a direct derivation from the pseudocode outlined in the source referenced here [1]. This resource encapsulates the foundational principles of the Boids algorithm, offering a comprehensive guide on how to implement each rule. The pseudocode serves as a blueprint for coding this model as well as act as a roadmap for understanding the complexities of the algorithm. Furthermore, this source extends the framework by introducing new rules and behaviors that can be easily integrated into the model. For example, it dives into the exploration and groundwork of the position bounding and speed limiting rules, both of which are discussed later in subsequent sections of this report.

This environment features 10 rovers ($N$), where each rover has an identification number ($i$). To determine the sequential movement for each rover, they undergo an update in their current velocity and position through the integration of three distinct steering velocity vectors - one for each rule. These vectors, each associated with the separation, alignment, and cohesion rules, collectively shape the behavior of the flock of rovers.

To tailor the collective behavior of these rovers, each steering velocity vector is assigned a weight that controls its influences on the individual rovers. This mechanic allows for the fine-tuning of the rovers' interactions with each other and their environment, allowing for the adjustment of the proximity and coordination levels among them. For example, assigning a higher weight to the cohesion rule encourages the rovers to maintain closer proximity, which fosters a greater unity within the flock. On the other hand, prioritizing the separation rule would cause the rovers to actively avoid each other, promoting a more spaced-out movement within the group.

*1) Rover Position:* Each rover is characterized by a position vector, shown in Equation 1, that represents its precise location in a 2-Dimensional space. Here, the horizontal ($x_i$) and vertical ($y_i$) variables are coordinates that define the rover's position in the environment. These vectors play a crucial role in tracking and managing the spatial distribution of rovers as they each navigate through the environment.

$$P_i = (x_i, y_i) \tag{1}$$

Updating the rovers' positions involves adding their corresponding velocities to their current positions as shown in Equation 2.

$$P_i = P_i + V_i \tag{2}$$

*2) Rover Velocity:* Similar to the position vector, each rover has a velocity vector, as seen in Equation 3. This vector encapsulates the speed and direction of the rover's movement. Specifically, the variable, $V_{x_i}$, indicates a rover's velocity in the horizontal ($x$) direction whereas the $V_{y_i}$ variable represents a rover's velocity in the vertical ($y$) direction. Together, these components will determine how fast the rover will move and in which direction it is heading at any given moment.

$$V_i = (V_{x_i}, V_{y_i}) \tag{3}$$

To dynamically adapt to the collective behavior modeled by the Boids algorithm, the velocities of each rover will undergo continuous adjustments. The three primary velocities that influences this adaptation are derived from the separation, alignment, and cohesion rules, all of which are explained later in this report. These velocities are added to the rover's current velocity vector, ensuring that this update allows the rover to respond to its environment instantly.

In a subsequent section of this report, an additional velocity will be introduced. This new velocity serves as a controller for the model, offering an additional layer of influence on the rover's movement. Unlike the defined velocities, this controller velocity will provide a higher-level of influence, guiding the rover's behavior with respect to the constraints within the environment.

*3) Separation Rule:* The separation rule within the Boids algorithm serves as a foundational principle dictating the actions of individual agents within a system. This rule plays a critical role in shaping the emerging patterns observed in the agents' collective motion, contributing to the realism of the simulation. The objective of this rule is to prevent collisions and maintain a safe distance between neighboring agents, mirroring the inherent behavior that is observed in

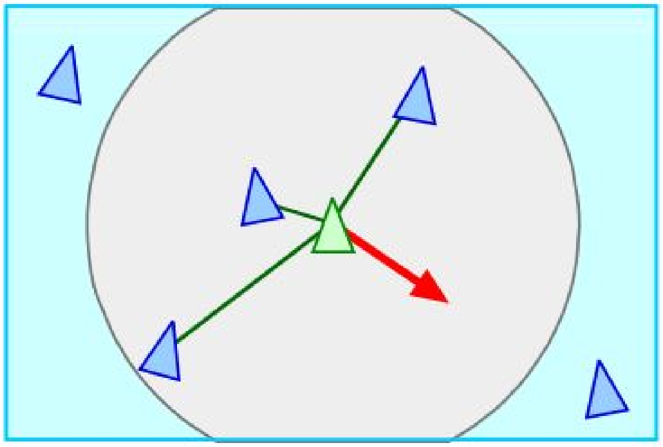

Fig. 3: Visualization of the separation rule where the green agent steers to avoid crowding local flockmates [2].

natural animal flocks [3]. By replicating this animal instinct, as illustrated in Figure 3, to avoid close distances to their peers, the rule ensures a well-spaced movement among the agents in the flock. Mimicking this natural awareness is relevant in the context of the Martian rovers, where having them maintain a safe distance with one another is essential to prevent collisions, enhance their navigational efficiency, and increase the chances of succeeding their given mission.

When executing this rule inside the algorithm, each rover will continuously check its surroundings using an adjustable sensing range called $R_{separation}$. This range serves as a radial zone which the rover uses to help detect the presence of other rovers. To determine if the adjacent rovers are inside this zone, the rover calculates the distance between itself and the other rovers using the Euclidean distance, shown in Equation 4. The Euclidean distance is the length of a line segment (or distance) between two points in 2-Dimensional space, which is calculated from the Cartesian coordinates of the points using the Pythagorean theorem. In this case, $(x_i, y_i)$ and $(x_j, y_j)$ are the respective position coordinates of rovers $i$ and $j$ in the 2D space.

$$d(P_i, P_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{4}$$

When the Euclidean distance between the rover and its neighbor falls below the $R_{separation}$ threshold, the rover will include its neighbor in its list that is used to generate the avoidance steering vector by using Equation 5. The list $(X)$ represents an array of rover $i$'s nearby neighbors. This list is pivotal for ensuring that the separation behavior is effectively applied to ignore other rovers that are not in the vicinity.

$$j \in X \iff (d(P_i, P_j) < R_{separation}) \tag{5}$$

Using the equations defined in this section, the separation velocity vector is calculated:

$$V_{separation} = a \times (\sum_{j \in X, j \neq i}^{N} (P_i - P_j)) \tag{6}$$

The scalar factor $(a)$ placed in front of the summation, is a tunable parameter that directs the weight or influence of the rover's separation behavior. This adjustable parameter will determine the strength of the rover's response to the separation steering vector in relation to the other influential behaviors in the algorithm. During the summation, which iterates over all rovers in the environment and only calculates neighbors stored inside list $X$, the exclusion of rover $i$ prevents accidental self-calculations where rover $i$ calculates with itself. This self-calculation can potentially introduce errors in the overall behavior of the system. Each term in the summation represents the positional difference between rover $i$ $(P_i)$ and its nearby neighbor, rover $j$ $(P_j)$. The summation computes the sum of these positional differences, and the result is then multiplied by the scalar factor, resulting the separation steering vector $(V_{separation})$.

$$V_i = V_i + V_{separation} \tag{7}$$

After the calculation of the separation velocity, the next step is to integrate this newly computed velocity with the rover's existing velocity. This process is crucial for determining the rover's updated movement direction and speed. Mathematically, this integration is represented as Equation 7. The addition of $V_{separation}$ onto $V_i$ controls the rover's overall movement by steering it away from nearby neighbors to prevent collisions and maintain a safe separation distance.

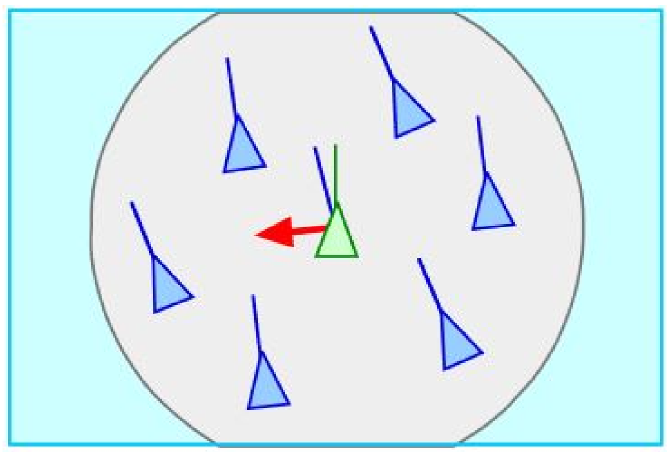

Fig. 4: Visualization of the alignment rule where the green agent steers towards the average heading of local flockmates [2].

*4) Alignment Rule:* Another fundamental component of the Boids algorithm is the alignment rule, a rule that plays a pivotal role in creating synchronization among individual agents. Its purpose involves changing the velocity of each agent within a flock to the calculated average velocity of their immediate neighbors, depicted in Figure 4 [3]. This idea is rooted in the concept of creating a harmonized movement within the flock, reflecting the dynamics observed in natural group behaviors.

For Martian rovers, this rule is of particular importance as it contributes to the overall efficiency and coordination of the multi-robot system. By having each rover align their velocities with their nearby peers, the rovers simulate a form of collective intelligence, improving their ability to navigate the defined Martian terrain in a dynamic and decentralized manner.

Much like the separation rule, the alignment rule relies on local interactions. Each rover sustains an awareness of its surroundings through a defined sensing range denoted as $R_{alignment}$. This range serves as a boundary within which the rover detects nearby peers, ensuring that the influence of each rover is only based on its immediate neighbors and the alignment calculations are accurate. This is depicted in Equation 8 below:

$$j \in X \iff (d(P_i, P_j) < R_{alignment}) \tag{8}$$

For each neighboring rover detected within $R_{alignment}$, the rover initiates the calculation of the average velocity of its peers, as expressed in Equation 9. This procedure involves summing up the individual velocities of each detected neighbor ($V_j$) and then dividing the total by the number of neighbors ($Y$). The resulting velocity provides a representation of the average direction and speed of the flock, a key parameter for guiding the rover's alignment.

$$V_{average} = \frac{1}{Y} \times (\sum_{j \in X, j \neq i}^{N} (V_j)) \tag{9}$$

To synchronize its movement with the flock's average velocity, the rover goes through a critical process of adjusting its own velocity in Equation 10. This adjustment is conducted by subtracting the rover's current velocity ($V_i$) from the average velocity of the flock ($V_{average}$). In situations where the rover fails to detect any peers within $R_{alignment}$, the new velocity ($V_{new}$) will return 0, indicating that the rover is not in proximity to any flocks or rovers in the environment.

$$V_{new} = \begin{cases} V_{average} - V_i & Y > 0 \\ 0 & Y \leq 0 \end{cases} \tag{10}$$

After the rover's velocity has been successfully adjusted, a scalar factor ($b$) is introduced in Equation 11. This variable is a changeable parameter, similar to $a$ in the separation rule, that determines the strength of the alignment behavior relative to the other behaviors within the Boids algorithm. This fine-tuning helps to achieve a synchronized movement within the flock, where each rover calibrates its trajectory with the collective direction of its close peers.

$$V_{alignment} = b \times V_{new} \tag{11}$$

Once the alignment velocity ($V_{alignment}$) is calculated, it gets integrated onto the rover's existing velocity, as illustrated in Equation 12. This integration process is significant in determining the rover's updated movement direction and speed, adjusting it with the calculated average trajectory of its nearby peers, helping in fostering the observed synchronized behavior within the system.

$$V_i = V_i + V_{alignment} \tag{12}$$

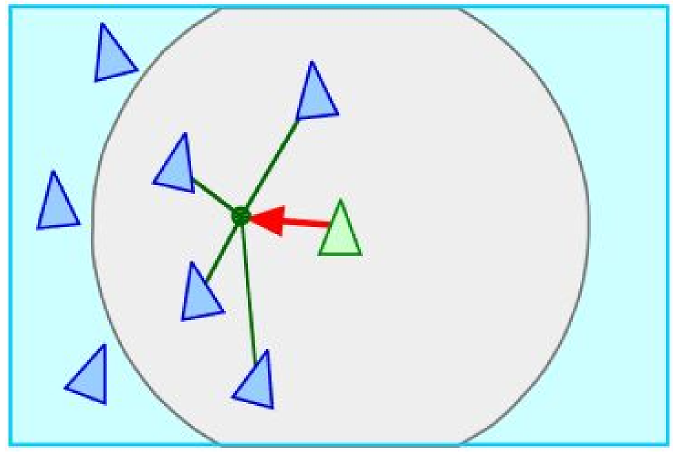

Fig. 5: Visualization of the cohesion rule where the green agent steers to move towards the average position of local flockmates [2].

*5) Cohesion Rule:* As the final aspect of the Boids algorithm, the cohesion rule plays a crucial role in making a collective sense of togetherness and unity within a flock of agents. Its objective is to create a harmonious movement among the agents by having each member of the flock move towards the calculated center of mass for the flock, depicted in Figure 5 [3]. This center of mass is a focal point that manages the coordinated movements of the agents, resulting in a united behavior within the system.

In the context of Martian rovers, this rule is of utmost importance. By encouraging the rovers to gravitate towards the center, the cohesion rule enhances the overall unity of the group. This coordination helps contribute to efficient data collection and fosters a unified front in navigating the Martian terrain, which is essential for the success of their mission.

Similar to its counterparts, the separation and alignment rules, the cohesion rule uses local interactions to shape the coordinated behavior of the rovers. Within this rule, each rover actively evaluates the positions of adjacent rovers within its designated sensing range, called $R_{cohesion}$. This range defines the spatial boundary within which the rover considers the presence of its neighboring peers. Equation 13 outlines how the rover includes every nearby rover within this range for cohesion calculations.

$$j \in X \iff (d(P_i, P_j) < R_{cohesion}) \tag{13}$$

The rover computes the average position of its detected neighbors within $R_{cohesion}$, as articulated in Equation 14. This process involves summing up the individual positions of each neighboring rover ($P_j$) and subsequently dividing the total by the number of neighbors. The result of this calculation signifies the positional representation of the flock's center of mass ($Mass_{center}$).

$$Mass_{center} = \frac{1}{Y} \times (\sum_{j \in X, j \neq i}^{N} (P_j)) \tag{14}$$

Following the calculation of the center of mass, the rover undergoes adjustments in its position, detailed in Equation 15. This computation entails subtracting the rover's current position ($P_i$) from the flock's center of mass. The result of this operation provides the new position ($P_{new}$) for the rover. In cases where the rover finds itself to be distant from any flocks or rovers, the new position assumes a value of 0, signifying that there is no center of mass for the rover to head towards.

$$P_{new} = \begin{cases} Mass_{center} - P_i & Y > 0 \\ 0 & Y \leq 0 \end{cases} \quad (15)$$

The succeeding step involves multiplying the rover's new position by a scalar factor ($c$), as depicted in Equation 16. Much like its counterparts, this scalar factor is an adjustable variable that determines the strength of the cohesion behavior relative to other behaviors within the algorithm.

$$V_{cohesion} = c \times P_{new} \quad (16)$$

The culmination of the cohesion rule then involves integrating the cohesion velocity ($V_{cohesion}$) into the rover's existing velocity. This process, represented by Equation 17, directs the rover to consistently move towards the flock's center, becoming a crucial factor in promoting union in flocks.

$$V_i = V_i + V_{cohesion} \quad (17)$$

### *D. Self-Defined Behaviors*

In this section, new behaviors and rules are added to the mathematical model described in Section II-C.

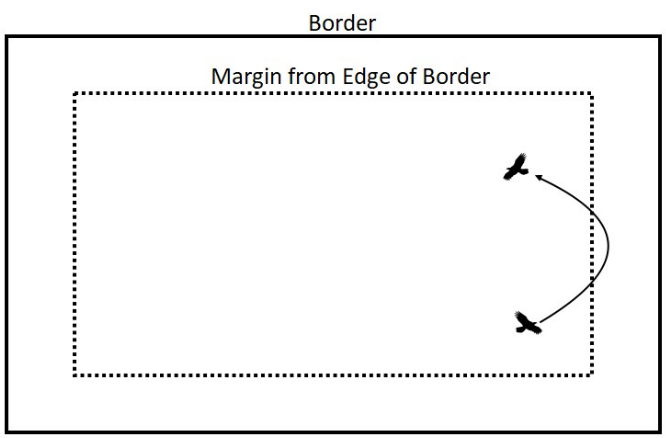


Fig. 6: Visualization of the border avoidance rule where the agent changes their trajectory once they reach the border margins [11].

*1) Border Avoidance:* The implementation of a border avoidance behavior in the Boids algorithm is critical for maintaining a realistic and visually coherent flocking simulation. In the context of this project, where rovers are simulating a Martian exploration mission, this behavior serves a dual purpose. First, it ensures that the rovers adhere to the defined Martian environment, illustrated in Figure 6, preventing them from drifting beyond the boundaries set [1]. This adherence not only contributes to the realism of the simulation but also aligns with the practical constraints that real Martian rovers face during their missions. Additionally, straying too far from the exploration area could potentially lead to the loss of communication links with the control center, hindering the rover's ability to transmit valuable data back to Earth and receive instructions. Therefore, this behavior acts as a safeguard that prevents the rovers from unintentionally wandering off and jeopardizing the mission's success.

The calculation process behind this behavior begins by systematically evaluating the rover's position ($P_x, P_y$) against each border margin ($M_{west}, M_{east}, M_{north}, M_{south}$), an area that signals the rover to initiate velocity adjustments before reaching the actual border. The assessment, detailed in Equation 18, prompts tweaks to the rover's horizontal velocity ($V_{x_i}$) if its position surpasses the western or eastern border margins. This adjustment is dictated by the predefined border trajectory ($T_{border}$).

$$V_{border_x} = \begin{cases} V_{x_i} + T_{border} & P_x < M_{west} \\ V_{x_i} - T_{border} & P_x > M_{east} \end{cases} \quad (18)$$

Simultaneously, the behavior assesses the rover's vertical position in Equation 19. If the rover goes beyond the northern or southern border margins, a correction is conducted on the rover's vertical velocity ($V_{y_i}$). Similar to the horizontal process, the vertical process is guided by the same border trajectory. These alterations represents the behavior's proactive response to the rover's proximity to the defined borders, aiming to maintain controlled movement within the set boundaries.

$$V_{border_y} = \begin{cases} V_{y_i} + T_{border} & P_y < M_{north} \\ V_{y_i} - T_{border} & P_y > M_{south} \end{cases} \quad (19)$$

After determining the horizontal and vertical border velocities, they are encapsulated into a vector, as shown in Equation 20. This vector captures the corrected velocity needed to help guide the rover away from the borders.

$$V_{border} = (V_{border_x}, V_{border_y}) \quad (20)$$

Equation 21 showcases the final step of this behavior's process, where the rover's current velocity gets updated. The border velocity is integrated into the rover's velocity, effectively controlling its steering behavior. This ensures that the rover responds accordingly to the new trajectory, reinforcing the chances of a successful exploration mission.

$$V_i = V_i + V_{border} \quad (21)$$

*2) Speed Limit:* Implementing a speed limit mechanism within the Boids algorithm is a strategic choice aimed at making the behavior of the simulated rovers be more realistic. This limit prevents the rovers' velocities from exceeding certain speeds, stopping any erratic and unrealistic fluctuations in their movement [1]. Without this constraint, the rovers would most likely reach exceptionally high speeds. Considering the realism rooted in natural animal movement, it is reasonable to incorporate this limitation due to the fact that real animals such as birds cannot move arbitrarily fast.


Fig. 7: Speed limit sign [6].

In the case of these Martian rovers, the speed limit ensures that they do not move at high speeds and risk depleting their power supplies quickly, jeopardizing the success of the mission. On the other hand, having them move too slowly leads to wasting valuable time, hindering their abilities to cover the designated exploration area within a desired time frame. Constraining the rovers to a set speed range emulates the constraints actual rovers face during their missions, where managing energy consumption and time efficiency are critical factors in mission completion.

The mathematical representation of the speed limit behavior is illustrated in Equation 22. This equation uses conditional statements to dynamically change the rover's velocity based on predefined limits. First, the magnitude of the rover's velocity ($|V_i|$) is compared in relation to the maximum velocity ($V_{max}$). If the magnitude is greater than the maximum velocity, then the maximum velocity gets multiplied by the rover's normalized velocity vector. This ensures that the rover's velocity decreases down to the maximum speed. Additionally, the magnitude is compared with the minimum velocity ($V_{min}$). If the magnitude is less than the minimum velocity, then the minimum velocity gets multiplied by the rover's normalized velocity vector, ensuring that the rover's velocity increases up to the minimum speed. This approach enforces the rover to have its speed remain within the designated speed range.

$$V_{speed} = \begin{cases} V_{max} \times (\frac{V_i}{|V_i|}) & |V_i| > V_{max} \\ V_{min} \times (\frac{V_i}{|V_i|}) & |V_i| < V_{min} \end{cases} \tag{22}$$

The resulting velocity ($V_{speed}$) is then updated onto the rover's current velocity, applying the defined speed constraints on it.

$$V_i = V_{speed} \tag{23}$$

## III. Theoretical Analysis

A theoretical analysis is performed on the mathematical model, where it explores the role of a controller in achieving the desired collective behavior of the Martian rovers. Additionally, the analysis dives into the mechanics implemented to guide the rovers towards the model's equilibrium state, characterized by the minimized potential function.

### A. Obstacle Avoidance

The obstacle avoidance rule, created for the Boids algorithm, plays an essential role in ensuring the navigation and safety of the Martian rovers during their exploration mission as well as function as a controller for the model. As the rovers traverse the Martian terrain, the real-time detection and response to obstacles are essential for ensuring their safety. This behavior, inspired by the separation rule, is strategically designed to prevent collisions and guide the rovers away from obstacles within their environment, as depicted in Figure 8 [4]. This mechanic emulates the natural tendency of animals avoiding physical barriers and aims to replicate a similar capability in this system.

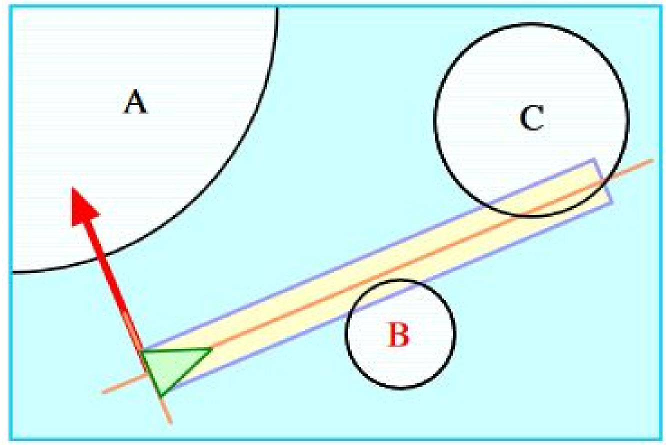


Fig. 8: Visualization of the obstacle avoidance rule where the green agent steers to move away from obstacles $B$ and $C$ [4].

Within the environment, 5 obstacles ($M$) are distributed randomly. Each rover in the system performs an assessment of the distance between itself and the obstacles ($o$). The evaluation involves determining the Euclidean distance between the rover's current position ($P_i$) and the position of each obstacle ($P_o$), illustrated in Equation 24. If this calculated distance falls below a predetermined range ($R_{obstacle}$), then it indicates that an obstacle is in close proximity of the rover [7]. Afterwards, the obstacle is stored in a list of nearby obstacles ($Z$), ensuring that this controller captures the potential obstacles that could hinder the rover's navigation.

$$o \in Z \iff (d(P_i, P_o) < R_{obstacle}) \tag{24}$$

Once the potential obstacles are identified, the controller then performs a summation over all obstacles inside $Z$, as shown in Equation 25. During this calculation, the vector difference between the rover's position and the obstacle's position is computed. This vector encapsulates the collective avoidance direction, accounting for the spatial distances of all detected obstacles. A scalar factor ($d$) is introduced to help determine the strength of this rule, acting as a control parameter and determining the influence the avoidance behavior will have relative to the other behaviors in the algorithm.

$$V_{obstacle} = d \times (\sum_{o \in Z}^{M} (P_i - P_o)) \tag{25}$$

Finally, the obstacle avoidance velocity ($V_{obstacle}$) then gets integrated into the rover's existing velocity, demonstrated in

Equation 26. By adding this avoidance velocity to the rover's current movement, the algorithm dynamically changes the rover's trajectory, allowing it to circumvent obstacles and maintain a safe navigation path.

$$V_i = V_i + V_{obstacle} \tag{26}$$

The obstacle avoidance rule acts as a real-time controller, dynamically steering the rovers to collectively respond to the environmental constraints posed by the obstacles. This integration not only enhances the realism of the simulation but also aligns with the objective of achieving a real Martian exploration mission.

### *B. Model's Equilibrium State*

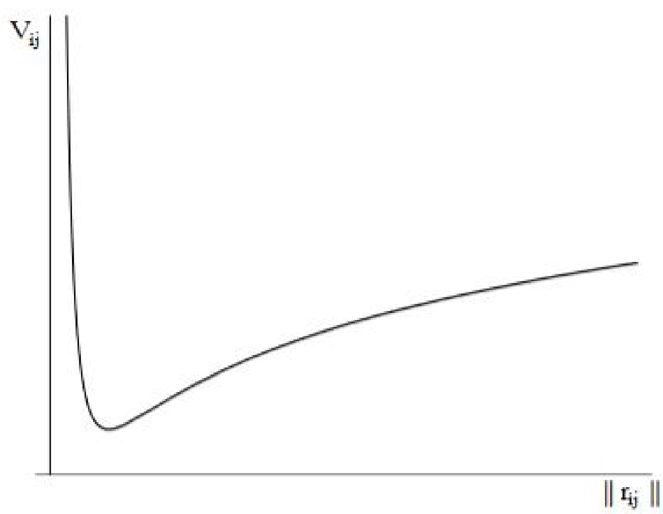


Fig. 9: Example of an inter-agent potential function [5].

The mathematical model of this system can be proven to have an equilibrium state using Definition IV.3 from the "Stable Flocking of Mobile Agents, Part I: Fixed Topology" paper by Tanner, Jadbabaie, and Pappas [5]. This paper explains that cohesion and separation forces exerted to a pair of neighboring robots are generated by a potential function ($V_{ij}$) in a way that satisfies this definition. An example of this is illustrated in Figure 9.

The definition states that the potential function is a differentiable, nonnegative, and radially unbounded function of the distance between robots $i$ and $j$ ($||r_{ij}||$) such that:

1) The potential function of the distance between the robots approaches $\infty$ as the distance approaches 0.
2) The distance between the robots approaches a unique value that minimizes the potentials of all robots.

The Boids algorithm incorporates a potential function that embodies the three fundamental rules: separation, alignment, and cohesion, shaping the collective behavior of the rovers in the system. The separation rule (Equation 6) enforces a strong repulsive force, preventing collisions and fostering a safe space among neighboring rovers. Simultaneously, the alignment rule (Equation 11) works towards synchronizing the velocities of the rovers in a flock, ensuring a coordinated movement. Finally, the cohesion rule (Equation 16) outputs an attracting force, pulling the rovers closer together and maintaining the overall structure of the flock [5]. The combination of these rules and the self-defined rules and behaviors, discussed earlier in the report, in the potential function reflects the interactions between these forces, collectively steering the system towards configurations that minimizes the rovers' potentials as time approaches $\infty$.

This potential function can then be mathematically expressed as the total potential of the rover, represented as:

$$V_i = \sum_{j \in X}^{N} V_{ij}(||r_{ij}||) \tag{27}$$

It is essential to note that, in the Boids algorithm, the distances between the rovers and their neighbors dynamically evolve over time. Unlike traditional static equilibrium states, the nature of this mathematical model introduces a dynamic aspect, ensuring that the equilibrium state is continuously changing and adapting to the evolving interactions among the rovers.

In addition to the foundational rules that constitute the core potential function, driving forces introduced by rules like obstacle avoidance (Equation 25) and border avoidance (Equations 18 and 19) play a significant role in shaping the behavior of the robots. These rules causes the rovers to momentarily leave the equilibrium state, prompting evasive maneuvers as they interact with their environment. For example, when confronted with obstacles in their paths, the rovers instantly change their trajectories to prevent collisions, causing them to disrupt the flock's equilibrium. These deviations highlight the non-static nature of the equilibrium state. Fortunately, the core Boids rules, guided by the potential function, efficiently work the system back towards a state of minimal potential, demonstrating the model's adaptability.

While the theoretical idea of achieving an equilibrium state involves the entire system reaching a configuration where all robots have their potentials minimized, practical considerations led to a nuanced understanding. In the context of this multi-robot system, the equilibrium state manifests when individual flocks, comprised of two or more rovers, minimize their potentials. Due to the decentralized nature of the Boids algorithm, the emergence of multiple flocks is a common occurrence. Each flock, characterized by minimized potentials among its flock members, achieves a local balance or stability even as the overall system remains dynamic [10]. Therefore, the equilibrium state is effectively realized when distinct flocks form, navigate, and maintain their configurations harmoniously. This approach highlights that local equilibrium is achieved independently within each flock, which contributes to the adaptive, dynamic nature of the entire system.

## IV. Validation in Simulations and/or Experiments

The validation of the proposed multi-robot system is crucial in assessing its effectiveness and applicability to the real-world. This section will present screenshots of these simulations and plots designed to validate properties derived from the Boids algorithm. The simulations of the robots are performed in Python using the *p5* library to animate the system. These simulations are conducted in order to validate the properties of the mathematical model described in Section III.

To help understand the interpretation of the screenshots and plots, the following list is created to showcase the meaning behind the various shapes, lines, and colors:

1) The colored triangles represent **rovers**.
2) The orange trails signify the rovers' **previous positions**.
3) The red circles are the **environmental obstacles**.
4) The colored lines represent the rovers' **velocities**.

Each of these items in the list are demonstrated in the Figures inside this section.

### A. *Table of Adjustable Parameters*

Building upon the discussion of adjustable parameters in the preceding sections, a table is presented encompassing all the key parameters pivotal to the model. Each of these parameters holds a distinct role in shaping the collective behavior of the system and their impact on the emerging patterns of the flock.

| **Parameters** | **Variables** | **Values** |
|---|---|---|
| Separation Sensing Range | $R_{separation}$ | 50 |
| Alignment Sensing Range | $R_{alignment}$ | 80 |
| Cohesion Sensing Range | $R_{cohesion}$ | 80 |
| Obstacle Avoidance Sensing Range | $R_{obstacle}$ | 60 |
| Separation Weight | $a$ | 0.15 |
| Alignment Weight | $b$ | 0.1 |
| Cohesion Weight | $c$ | 0.005 |
| Obstacle Avoidance Weight | $d$ | 0.25 |
| Northern Border Margin | $M_{north}$ | 175 |
| Eastern Border Margin | $M_{east}$ | 1025 |
| Western Border Margin | $M_{west}$ | 175 |
| Southern Border Margin | $M_{south}$ | 1025 |
| Border Trajectory | $T_{border}$ | 10 |
| Maximum Speed | $V_{max}$ | 10 |
| Minimum Speed | $V_{min}$ | 7 |

TABLE I: Recorded Values of Each Adjustable Parameter

### B. *Obstacle Avoidance Results*

The obstacle avoidance rule was thoroughly examined through simulation, offering valuable insights into how Martian rovers navigate through their environment when faced with potential obstacles. This critical aspect of the simulation aims to demonstrate the efficiency of this rule and shed light on the model's ability to ensure the rovers adapt their trajectories accordingly to circumvent obstacles in their path.

A crucial moment captured in the simulation, as illustrated in Figure 10, showcases the obstacle avoidance property. In this particular instance, the purple rover (Rover #2) and the green-yellow rover (Rover #9) detect an obstacle within their sensing ranges, $R_{obstacle}$, prompting an immediate adjustment in their trajectories. This frame visually confirms that these rovers demonstrated this property as well as emphasize the rovers' abilities to autonomously alter their paths in response to potential collisions.

Figures 11 and 12 provides additional insights into the behaviors of this property. More specifically, these figures reveal distinct changes in the $Y$ velocities of these two rovers between $t = 3$ seconds and $t = 7$ seconds, corresponding to the moment they detected an obstacle, confirming that the rovers dynamically adapt their velocities when they encounter obstacles.

Through numerous simulation iterations, the rovers have consistently displayed their adaptability in response to various obstacles near them or in front of their paths. The model's obstacle avoidance rule successfully prevented collisions with the environment's obstacles by steering the rovers away from potential hazards. This adaptability is crucial, ensuring the rovers navigate through a diverse and unpredictable Martian terrain while they efficiently collect data.

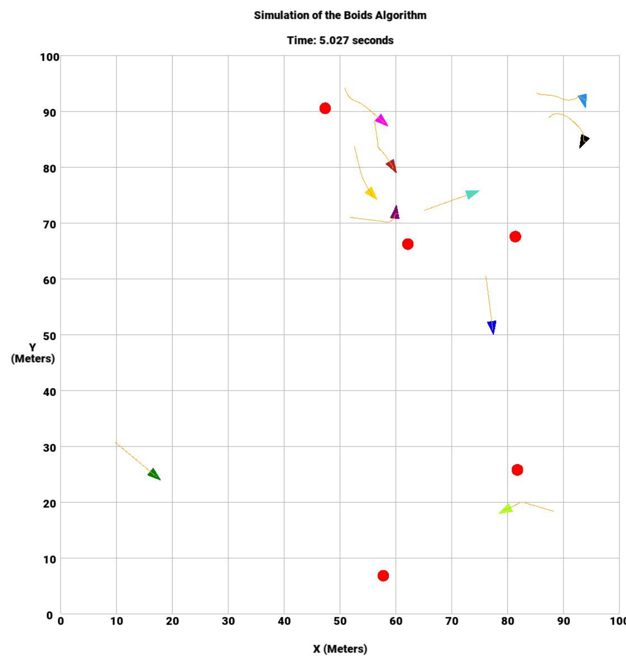


Fig. 10: The trajectories of all rovers at $t \approx 5$ seconds during the Trial 2 simulation.

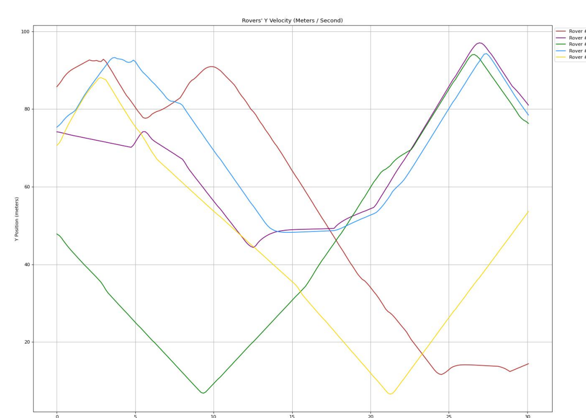


Fig. 11: The $Y$ velocities of Rovers 1 through 5 from the Trial 2 simulation.

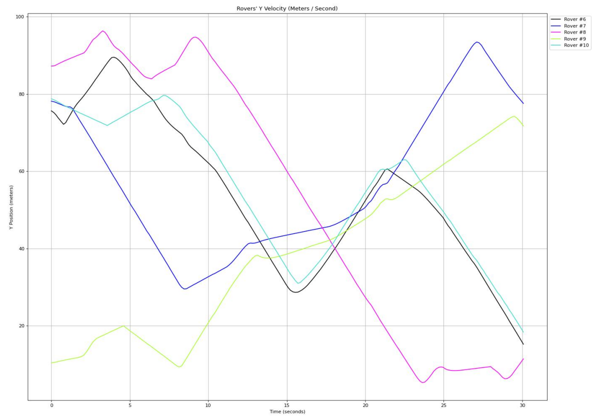


Fig. 12: The $Y$ velocities of Rovers 6 through 10 from the Trial 2 simulation.

### *C. Model's Equilibrium State Results*

The validation of the model's equilibrium state is a crucial component in the simulation analysis. In accordance to the properties discussed in Section III, the equilibrium state is characterized by the system approaching a configuration that minimizes the potential function. This function, constructed from the combination of the separation, alignment, and cohesion rules, adapts to the local interactions within each individual flock. It is crucial to emphasize that the potential function serves as a collective measure of the system's internal forces, guiding each flock to minimize their flock members' potentials based on their local interactions. This section aims to analyze the model's responsiveness to environmental interactions, focusing on the relationship between the local equilibrium and the responses to their surrounding environment.

Figure 13 captures a significant moment in the simulation, showcasing multiple flocks of rovers demonstrating the expected behavior of minimizing potentials within their respective flocks. This frame clarifies that, through the collaboration of the three rules, the rovers achieve a local configuration where they are minimally close to each other, reinforcing the model's capability to guide the flock toward an equilibrium state.

Figures 15 and 16 illustrates the $X$ and $Y$ velocities of the three rovers - the fire-brick rover (Rover $\#1$), the purple rover (Rover $\#2$), and the golden rover (Rover $\#5$) - captured during the Trial 1 simulation. Throughout the equilibrium state of this particular flock, observed between $t \approx 13$ seconds and $t \approx 22$ seconds, the $Y$ velocity plot showed a stable and synchronized movement between these rovers. Clearly demonstrated by the consistency of their velocity patterns, this synchronization aligns with the collective behavior obtained through the potential function. The rovers achieved a state of minimal potential during this period, highlighting the theoretical expectations of an equilibrium state.

Notably, the $X$ velocity plot revealed the active influence of the separation rule. As rovers in the flock approached each other, slight alterations in their trajectories were observed, signifying the effective implementation of this rule. This behavior helped ensure that the rovers maintained a safe and minimal distance from one another, contributing to the overall stability of the flock.

On the other hand, Figure 14 captures the moment when the flock leaves its equilibrium state due to its encounter with the environment's borders. Here, the border avoidance rule became relevant. The rule controlled the rovers' trajectories when they approached the area's borders, forcing them to deviate from their equilibrium state. This was crucial to ensure that the rovers stay within the environment, emphasizing the important balance of maintaining the flock's equilibrium and the need to respond to external factors.

In a concurrent observation, Figures 17 and 18 depicts the dynamics of another flock, consisting of the blue rover (Rover $\#7$) and the magenta rover (Rover $\#8$), during the same trial simulation. The $Y$ velocity plot in Figure 18 distinctly illustrates this flock's equilibrium state, which started at $t \approx 8$ seconds and lasted until the end of the simulation. Similar to the first flock, this plot indicated a harmonious movement showcasing this flock achieving a minimized potential state.

Subsequently, the $X$ velocity plot in Figure 17 reveals the active application of the separation rule within this flock. From $t \approx 9$ seconds until the end of the simulation, both rovers constantly adjusted their $X$ velocities. This behavior is a response to maintaining a safe distance from each other and thereby preventing collisions.

By comparing these two flocks, each with their own set of rovers, the simulation results captures the versatility and adaptability of this multi-robot system in achieving local equilibrium states while dynamically responding to their surrounding environment.

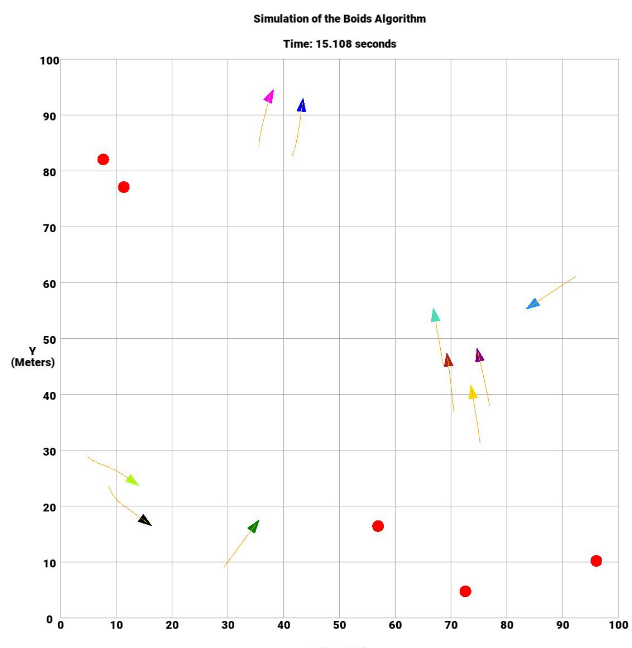


Fig. 13: The trajectories of all rovers at $t \approx 15$ seconds during the Trial 1 simulation.

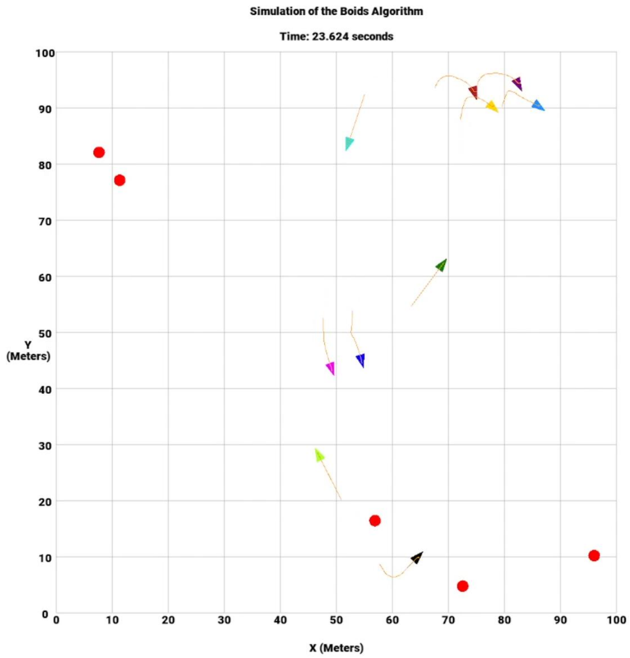


Fig. 14: The trajectories of all rovers at $t \approx 23$ seconds during the Trial 1 simulation.

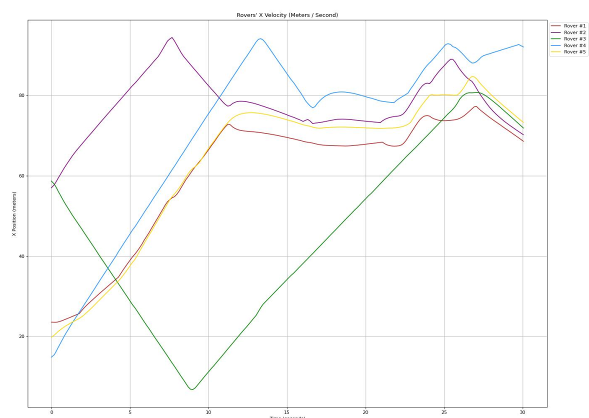


Fig. 15: The $X$ velocities of Rovers 1 through 5 from the Trial 1 simulation.

## V. Conclusion

The exploration and analysis of the multi-robot system governed by the Boids algorithm provided valuable insights into the behaviors and adaptability of the rovers in a simulated Martian environment. The study focused on various aspects, including the implementation of the fundamental rules of the Boids algorithm, the incorporation of additional rules such as border avoidance, speed limit, and obstacle avoidance, and the validation of the model properties through simulations.

The findings presented in this report holds significance for the development and deployment of multi-robot systems in real-world applications. The adaptability showcased by the

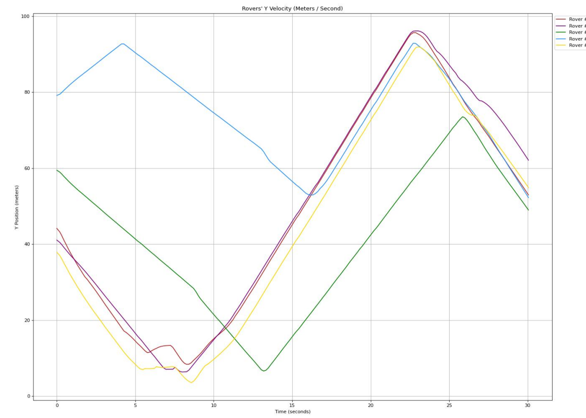


Fig. 16: The $Y$ velocities of Rovers 1 through 5 from the Trial 1 simulation.

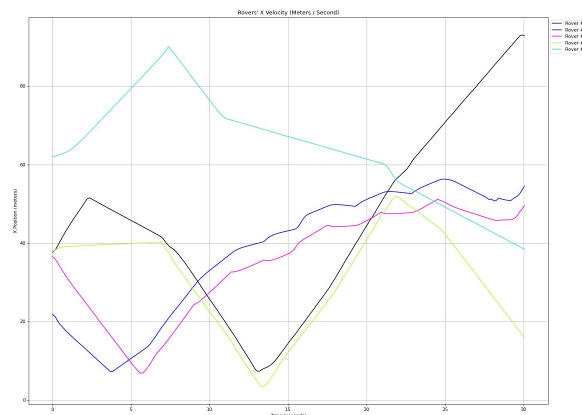


Fig. 17: The $X$ velocities of Rovers 6 through 10 from the Trial 1 simulation.

Boids algorithm, more specifically in response to obstacles and environmental constraints, suggests its potential application in scenarios where collective and dynamic behaviors are essential.

Future work could dive deeper into optimizing the rule parameters, investigate the impact of varying environmental complexities such as an uneven area and canyons, and extending the model to accommodate more intricate scenarios.

## VI. Appendix

Video links that demonstrate this multi-robot system are provided below:

- Trial 1
- Trial 2
- Trial 3

To ensure the project files work, two empty folders, *Plots* and *Screenshots*, are provided in the submission. This is where the plots and frames from the project simulations will be

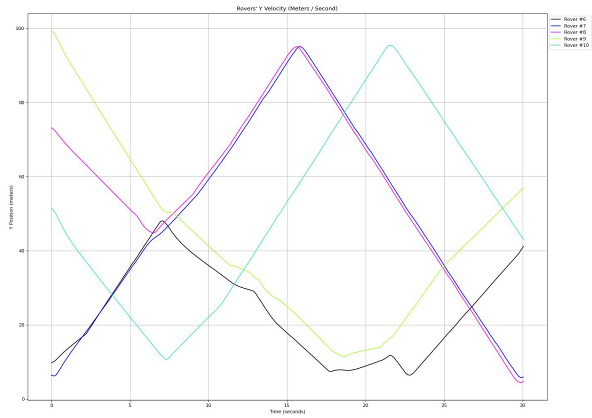


Fig. 18: The $Y$ velocities of Rovers 6 through 10 from the Trial 1 simulation.

stored once the simulation ends. Additionally, the plots and screenshots for each trial are also provided.

The two Python files, *main.py* and *boids.py*, must be in the same folder in order to run the simulation. Additionally, the *main.py* needs to be executed in order for the simulation to work. To ensure that no errors occur, make sure to have the *p5*, *numpy*, *math*, and *matplotlib* libraries installed in your Python IDE.

Lastly, there is a font file called *Roboto-Black.ttf* provided in the submission. This file is meant to have the text inside the simulation be that font style. Removing this file or moving to a different folder will break the simulation.

## References


[1] Conrad Parker, "Boids Pseudocode," *Verge Networks*, Sep. 06, 2007. https://vergenet.net/~conrad/boids/pseudocode.html.

[2] Craig W. Reynolds, "Boids (Flocks, Herds, and Schools: a Distributed Behavioral Model)," *Red3d*, Sep. 06, 2001. https://www.red3d.com/cwr/boids/.

[3] Craig W. Reynolds, "Flocks, Herds and Schools: A Distributed Behavioral Model," *ACM SIGGRAPH Computer Graphics*, vol. 21, no. 4, pp. 25–34, Aug. 1987, doi: https://dl.acm.org/doi/pdf/10.1145/37402.37406.

[4] Craig W. Reynolds, "Steering Behaviors For Autonomous Characters," *Sony Computer Entertainment America*, pp. 763-782, 1999. Available: https://www.red3d.com/cwr/papers/1999/gdc99steer.pdf.

[5] Herbert G. Tanner, Ali Jadbabaie, and George J. Pappas, "Stable Flocking of Mobile Agents, Part I: Fixed Topology," in *42nd IEEE International Conference on Decision and Control (IEEE Cat. No.03CH37475)*, Dec. 2003, pp. 2010–2015. doi: https://doi.org/10.1109/CDC.2003.1272910.

[6] Infinity Auto, "How to Stay Inside the Speed Limit," *Infinity Insurance*. https://www.infinityauto.com/knowledge-center/road-safety-advice/tips-keep-your-car-posted-speed-limit.

[7] Mahamad S. Yalagi, Pongupala P. Kumar, Manish Kumar, Malpani V. Rakesh, and Sailaja Thota, "Flocking with Distance Perception Obstacles Avoidance," *International Journal of Science, Engineering and Management (IJSEM)*, vol. 5, no. 6, Jun. 2020, Available: https://www.technoarete.org/common_abstract/pdf/IJSEM/v7/i6/Ext_73218.pdf.

[8] NASA, "Mars Curiosity Rover | Instruments," *NASA's Mars Exploration Program*, 2019. https://mars.nasa.gov/msl/spacecraft/instruments/summary/.

[9] NASA, "Mars Curiosity Rover | Curiosity Rover Mission Overview," *NASA's Mars Exploration Program*, 2019. https://mars.nasa.gov/msl/mission/overview/.

[10] Thierry Mora, Aleksandra M. Walczak, Lorenzo D. Castello, Francesco Ginelli, Stefania Melillo, Leonardo Parisi, Massimiliano Viale, Andrea Cavagna, and Irene Giardina, "Local Equilibrium in Bird Flocks," *Nature Physics*, vol. 12, no. 12, pp. 1153–1157, Aug. 2016, doi: https://doi.org/10.1038/nphys3846.

[11] Van H. Adams, "Boids algorithm - augmented for distributed consensus," *vanhunteradams.com*. https://vanhunteradams.com/Pico/Animal_Movement/Boids-algorithm.html.

[12] Wikipedia Contributors, "Flock (birds)," *Wikipedia*, Jul. 25, 2023. https://en.wikipedia.org/wiki/Flock_%28birds%29.