

# CLIPSwarm: Converting text into formations of robots

Pablo Pueyo, Eduardo Montijano, Ana C. Murillo and Mac Schwager

**Abstract**—We present CLIPSwarm, an algorithm to generate robot swarm formations from natural language descriptions. CLIPSwarm receives an input text and finds the position of the robots to form a shape that corresponds to the given text. To do so, we implement a variation of the Montecarlo particle filter to obtain a matching formation iteratively. In every iteration, we generate a set of new formations and evaluate their Clip Similarity with the given text, selecting the best formations according to this metric. This metric is obtained using Clip, [1], an existing foundation model trained to encode images and texts into vectors within a common latent space. The comparison between these vectors determines how likely the given text describes the shapes. Our initial proof of concept shows the potential of this solution to generate robot swarm formations just from natural language descriptions and demonstrates a novel application of foundation models, such as CLIP, in the field of multi-robot systems. In this first approach, we create formations using a Convex-Hull approach. Next steps include more robust and generic representation and optimization steps in the process of obtaining a suitable swarm formation.

## I. INTRODUCTION AND RELATED WORK

*Artistic robotics* has emerged as a promising field in recent years for both the general audience and the robotics community. The main focus of this trend is using robots to express or design art in all manners. For instance, some existing works aim to convert robots into painters, with techniques that go from copying existing techniques from real painters [2], [3], preprocessing the input images to simplify the input given to the robot [4] or even to paint graffiti autonomously [5]. In other works, the robots are able to sculpt sculptures without human interaction [6]. Other robotic works explore diverse expressions of art like dance [7] or autonomous drone cinematography. In this last stream, the robots are able to record cinematographic scenes autonomously satisfying some artistic or technical details [8], [9].

One of the latest streams of robotics arts is the use of a team of robots or drones that form creating artistic shapes, which is the main topic of this work. One proof of interest in this trend for the general audience is the growing number of companies that perform shows where a number of drones act as pixels and coordinate to form visually appealing shapes in the sky. The stream of multirobot 2D artistic robotic formations is

This work was supported by a DGA scholarship; DGA project T45\_23R, by MCIN/AEI/ERDF/European Union NextGenerationEU/PRTR project PID2021-125514NB-I00; NSF grants CNS-1330008 and IIS-1646921; ONR grants N00014-18-1-2830, and N62909-19-1-2027.

P. Pueyo, E. Montijano and A. C. Murillo are associated with the Instituto de Investigación en Ingeniería de Aragón, Universidad de Zaragoza, Spain {ppueyo, emonti, acm}@unizar.es

M. Schwager is associated with Dept. of Aeronautics and Astronautics, Stanford University, USA {schwager}@stanford.edu

Please cite this article as “P. Pueyo, E. Montijano, A. C. Murillo, and M. Schwager, *CLIPSwarm: Converting text into formations of robots*. ICRA 2023 Workshop on Multi-Robot Learning”

Description: “*The contour of a drop*”

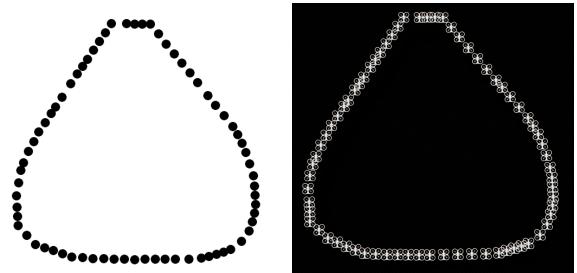


Figure 1. **Drone formation reproduced the given text.** This solution takes natural language descriptions as input and decides on the positions of a swarm formation of robots to correspond to the text. The example shows the shape created by a formation of 70 robots from above. On the left, a graphical representation of the shape, and on the right, the same formation represented in photorealistic simulation. The drones move to positions that form a shape that corresponds to the text “*The contour of a drop*”.

also addressed in the academic world. Some works focus on optimal control to move the robots forming a set of given patterns, [10], [11], even incorporating an interface to draw the desired pattern [12]. More recent approaches address how to perform multidrone 3D formations shows [13]–[15].

The aforementioned solutions need the interaction of a person to manually design the shape that the robots should form. CLIPSwarm paves the way and is the first step to creating these formations autonomously, which is the main contribution of this work. The user introduces a description in natural language of the desired shape of the formation, e.g. “*A shape of a hexagon*”, and CLIPSwarm decides the position of the robots of the formation to form a shape that corresponds with that description, so the users do not need to create the patterns beforehand.

To do so, we generate different drone formations iteratively following an adaptation of a Montercarlo particle filter. We select the formations with the highest similarity, improving the similarity between the created formations and the introduced text in each iteration. This similarity is given by CLIP [1], a foundation model that is trained to detect similarities between texts and images.

Specifically, CLIP encodes both the text and the images formed by the robot formations, extracting their similarity or matching likelihood that defines the possibility of the given text describing the given image. The particle filter selects and iterates over the formations that are more likely to describe the input text. At the end of the process, we obtain a resulting formation that best describes the input text based on its CLIP encoding. This configuration can be sent to a planning or formation algorithm to move the robots to the desired position, forming the input shape.

We validated the approach through different test cases and reproduced one of the formations in photorealistic simulation, demonstrating CLIPSwarm’s ability to create swarm forma-

tions of robots that correspond to given descriptions.

## II. SOLUTION

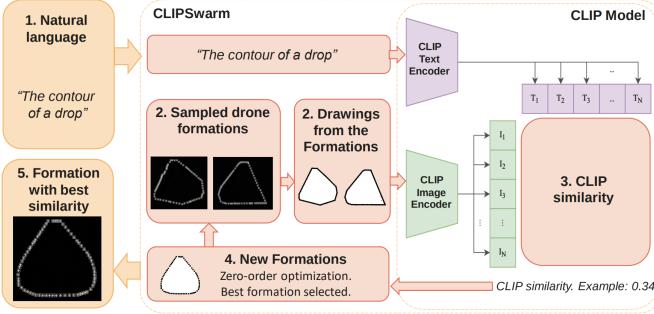


Figure 2. ClipSwarm diagram.

The diagram depicted in Fig. 2 shows a graphical representation of the algorithm implemented in CLIPSwarm. The input (1) is a description in natural language and the output (5) is a drone formation with the best similarity.

The next algorithm is an adaptation of the MonteCarlo algorithm [16] for particle filter robot localization and presents the different steps that CLIPSwarm implements (steps 2-4 in the diagram) to model the formations of robots that improve the similarity in each iteration:

---

### Algorithm 1 CLIPSwarm algorithm

---

**Require:**  $n > 0, m > 0, b \geq 0, k \geq 0, h \geq 0, r \geq 0$

- 1:  $\mathbf{F} \leftarrow \text{random}(n, m) + \text{predefined\_formations}()$
- 2: **for**  $it \neq 0$  **do**
- 3:      $images \leftarrow \text{draw\_formations}(\mathbf{F})$
- 4:      $scores \leftarrow \text{clip}(images)$
- 5:      $\mathbf{BF}, B \leftarrow \text{best}(\mathbf{F}, scores, b)$
- 6:      $\mathbf{NF} \leftarrow \text{get}(\mathbf{F} \propto scores, k)$
- 7:      $\mathbf{F} \leftarrow \mathbf{BF}$
- 8:      $\mathbf{F} \leftarrow \mathbf{F} + \text{alter\_best}(\mathbf{BF})$
- 9:      $\mathbf{F} \leftarrow \mathbf{F} + \text{random\_actions\_interior}(\mathbf{BF}, h)$
- 10:     $\mathbf{F} \leftarrow \mathbf{F} + \text{random\_actions}(\mathbf{NF})$
- 11:     $\mathbf{F} \leftarrow \mathbf{F} + \text{random}(r, m)$
- 12:     $it \leftarrow it - 1$
- 13: **end for**

---

**1.  $\mathbf{F} \leftarrow \text{random}(n, m) + \text{predefined\_formations}()$ :** In the first step of the algorithm the set of formations, which is represented by  $\mathbf{F}$ , is initialized randomly. The set is composed of  $n$  formations that are formed by  $m$  robots. This step uses a uniform distribution to initialize the 2D positions of the robots randomly between some limits. Additionally, in this first step, some formations are replaced with predefined formations (square, triangle, circle, inverted triangle, hexagon) and some random modifications of them to help the algorithm to 'warm start'.

**3.  $images \leftarrow \text{draw\_formations}(\mathbf{F})$ :** The positions of the robots are used to create images that represent the formations. The images are created by drawing a contour over the robots that are in the external part of the formation. We use a Convex-Hull algorithm [17] to calculate this contour and

give more information to CLIP, which is drawn with a black line (see examples in Fig. 2-Drawings from the formations). Additionally, this algorithm extracts the robots belonging to the contour and the interior of the shape to be used in step 9.

**4.  $scores \leftarrow \text{clip}(images)$ :** The formations are evaluated, detecting which ones are more likely to be described by the given description. For this purpose, we use a metric called *Clip Similarity*, and that is higher if a given text and image are likely to represent the same concept or idea. To create this metric, we use CLIP [1], a neural network solution that matches pairs of text and images. The network was trained with a big number of texts and images that are encoded into vectors.

In our solution, we first encode the images of the formations using CLIP, as well as the given text. The cosine similarity between the text and image vectors of each formation determines its *Clip Similarity Score*. The function  $\text{clip}(\cdot)$  returns the list of the scores for every formation.

**5.  $\mathbf{BF}, B \leftarrow \text{best}(\mathbf{F}, scores, b)$ :** The function  $\text{best}(\cdot)$  selects the  $b$  formations from  $\mathbf{F}$  with higher scores and the formation  $B$ , which is the formation with the highest score. The so-called *best-formations* are saved into the set  $\mathbf{BF}$  to be used in next iterations.

**6.  $\mathbf{NF} \leftarrow \text{get}(\mathbf{formations} \propto scores, k)$ :** In this step, we select  $k$  formations from  $\mathbf{F}$  proportionally to their score, so the ones that have higher scores are selected with higher probability.

**7.  $\mathbf{F} \leftarrow \mathbf{BF}$ :** The formation set  $\mathbf{F}$  used in the next iteration is initialized with the  $b$  best formations selected in step 5.

**8.  $\mathbf{F} \leftarrow \mathbf{F} + \text{alter\_best}(\mathbf{BF})$ :** In this step, we create  $bm$  new formations by altering the formations of  $\mathbf{BF}$ . For instance, we alter the position of one robot of the best formations each time by adding a random number  $\in (-lim, lim)$  to its coordinates. Additionally, we create new formations moving all the robots to the contour.

**9.  $\mathbf{F} \leftarrow \mathbf{F} + \text{random\_actions\_interior}(\mathbf{BF}, h)$ :** Similarly to the previous step and using the division from step 3, new formations are added to the new set. The new formations are created by randomly modifying the position of the robots that are inside of the hull of the formations of  $\mathbf{BF}$ ,  $h$  times. The robots that belong to the contour of the shape remain static.

**10.  $\mathbf{F} \leftarrow \mathbf{F} + \text{random\_actions}(\mathbf{NF})$ :** The robots of each formation selected in step 6 are moved with different actions. The coordinates of each robot are modified by adding a random number  $\in (-lim, lim)$ , and the resulting set is stored in  $\mathbf{F}$  for the next iteration of the algorithm.

**11.  $\mathbf{F} \leftarrow \mathbf{F} + \text{random}(r, m)$ :** Similarly to step 1, new  $r$  random formations, where  $m$  robots are distributed following a uniform distribution, complete the set  $\mathbf{F}$  set of the next iteration.

**2-12-13. New iteration:** The algorithm is repeated with the new formation as the initial point for  $it$  iterations. At the end of the algorithm, we select the formation  $B$  with the highest score which is the more likely to be described by the given text.

### III. EXPERIMENTAL VALIDATION

In this section, we demonstrate CLIPSwarm's ability to generate formations that correspond to a given natural language description. We first depict newly created formations from a set of descriptions. Then, we show a reproduction of the new formations in the photorealistic drone simulator AirSim to show a scenario closer to reality.

#### A. Modeling new formations

This section shows some examples of new formations created by the algorithm based on an input description and how they evolve through the iterations. The parameters used in the experimentation are listed in Table I. The algorithm runs for  $it = 50$  iterations, creating and comparing  $n = 1600$  formations, each consisting of  $m = 70$  robots, at each iteration. The results are displayed in Fig. 3. Each row shows the representation of the formation with the highest score at different iterations of the algorithm, illustrating the evolution of the shape until the last iteration for qualitative comparison. Each robot is represented with a black point. The last formation in each row represents the final positions of the robots, where the Clip Similarity reaches its maximum value, as visually demonstrated in the figure. For quantitative results, the plot in the last column of the figure illustrates the evolution of the Clip Similarity metric throughout the iterations. This metric increases as the iterations progress, indicating the solution's ability to select formations that best represent the given description. The line eventually stops increasing significantly, providing an indication of when to stop the process to save time and resources. Additionally, Table II shows the best Clip Similarity for each case for the initial and final iteration.

#### B. Formation with drones in photorealistic simulation

For qualitative results, we recreated the formations of the previous experiment with drones of the photorealistic simulator AirSim [18], [19], a scenario closer to reality. The connection between CLIPSwarm and AirSim is implemented in ROS [20], which would ease the transfer of the solution to a setup with real drones. The results are depicted in the fifth column of Fig. 3.

### IV. LIMITATIONS

For the sake of simplicity, we have decided to evaluate the formations based on the Clip Similarity of the Convex-Hull

Table I  
PARAMETERS

$lim$	$it$	$m$	$b$	$r$	$k$	$h$	$n$
2	100	70	40	230	250	5	$b+mb+r+k+hb = 1600$

Table II  
BEST SIMILARITY/ITERATION

	Case 1	Case 2	Case 3	Case 4
it=1	0.271	0.303	0.311	0.294
it=50	0.312	0.320	0.342	0.335

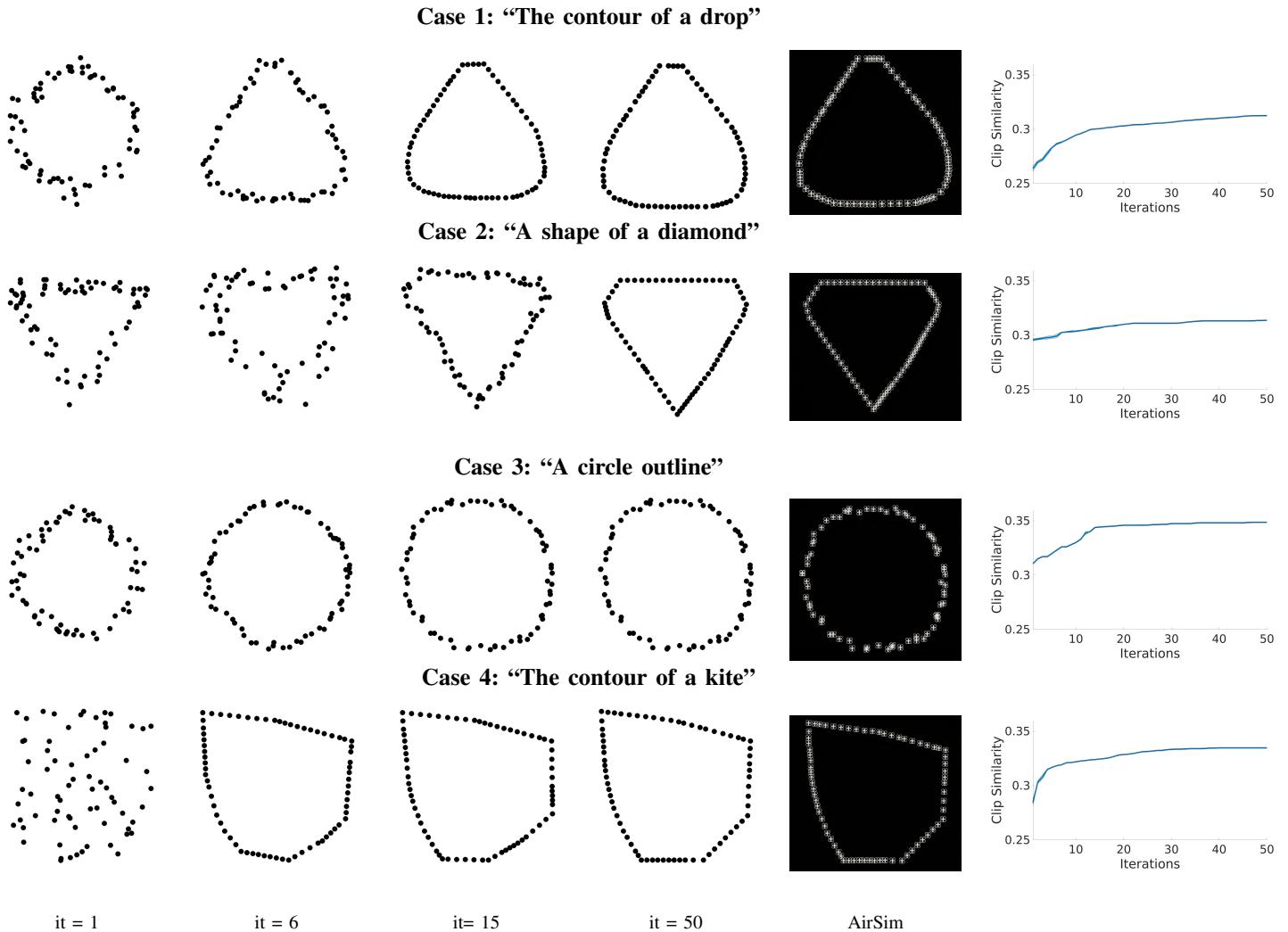
contour of the formations in this preliminary version. This simplification allows for a faster evaluation process and works well with a lower number of robots. However, the variety of shapes that can be modeled with a convex contour is limited. Moreover, this algorithm relies significantly on Clip, and increasing the similarity does not always mean that the shape is closer to what an average user would expect. We illustrate this with an example in Fig. 4, where the Convex-Hull fails to capture all the expected details that would represent a house. In this case, the Clip Similarity of the picture on the left is much higher than the one on the right, even though the latter may seem closer to a *house* for an average reader. Future steps will include working with more complex inputs as well as using additional metrics that complement the Clip Similarity for a more insightful comparison of the images and texts.

### V. CONCLUSIONS

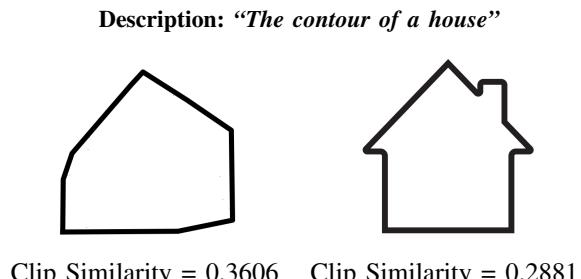
In this paper, we presented CLIPSwarm, an algorithm for modeling robot swarm formations to represent a given natural language description automatically. We described an iterative algorithm that is an adaptation of a Montecarlo particle filter to obtain the formation of robots that is more likely to be described by the given text. To measure the similarity between the description and the image of the formations, we used CLIP to encode the text and the images into vectors and obtain their similarity. Our experiments demonstrated the system's ability to model robot formations from natural language descriptions, and its potential implementation in a platform of drones using ROS. This is a first step towards the problem, and future work will explore a wider variety of formations, including more complete shapes and 3D formations. Furthermore, we aim to improve the process of moving the drones to the final positions with more sophisticated planning algorithms.

### REFERENCES

- [1] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *Int. Conf. on Machine Learning*, 2021, pp. 8748–8763.
- [2] L. Scalera, S. Seriani, A. Gasparetto, and P. Gallina, "Non-photorealistic rendering techniques for artistic robotic painting," *Robotics*, vol. 8, no. 1, p. 10, 2019.
- [3] A. Beltramello, L. Scalera, S. Seriani, and P. Gallina, "Artistic robotic painting using the palette knife technique," *Robotics*, vol. 9, no. 1, 2020.
- [4] A. Karimov, E. Kopets, G. Kolev, S. Leonov, L. Scalera, and D. Butusov, "Image preprocessing for artistic robotic painting," *Inventions*, vol. 6, no. 1, p. 19, 2021.
- [5] G. Chen, S. Baek, J.-D. Florez, W. Qian, S.-w. Leigh, S. Hutchinson, and F. Dellaert, "Gtgraffiti: Spray painting graffiti art from human painting motions with a cable driven parallel robot," in *Int. Conf. on Robotics and Automation*, 2022, pp. 4065–4072.
- [6] Z. Ma, S. Duenser, C. Schumacher, R. Rust, M. Baecher, F. Gramazio, M. Kohler, and S. Coros, "Stylized robotic clay sculpting," *Computers & Graphics*, vol. 98, pp. 150–164, 2021.
- [7] H. Peng, C. Zhou, H. Hu, F. Chao, and J. Li, "Robotic dance in social robotics—a taxonomy," *IEEE Transactions on Human-Machine Systems*, vol. 45, no. 3, pp. 281–293, 2015.
- [8] R. Bonatti, W. Wang, C. Ho, A. Ahuja, M. Gschwindt, E. Camci, E. Kayacan, S. Choudhury, and S. Scherer, "Autonomous aerial cinematography in unstructured environments with learned artistic decision-making," *Journal of Field Robotics*, vol. 37, no. 4, pp. 606–641, 2020.
- [9] P. Pueyo, E. Montjano, A. C. Murillo, and M. Schwager, "Cinempc: Controlling camera intrinsics and extrinsics for autonomous cinematography," in *Int. Conf. on Robotics and Automation*, 2022, pp. 4058–4064.



**Figure 3. Modeling formations from texts** First four columns: Formations of 70 robots with the highest similarity in the different iterations (1, 6, 15, 50) of the algorithm given the description detailed in the title of each row. The fourth column represents the formation with the highest similarity after 50 iterations. This formation is reproduced with simulated drones in AirSim and depicted in the fifth column. Last column: Evolution of the Clip Similarity along the iterations. The Clip Similarity increases significantly in the first iterations.



**Figure 4. Limitations of the algorithm.** This example illustrates the limitations of the algorithm. The Convex-Hull simplifies the evaluation process but may not capture all the details of the formation's shape. Additionally, relying solely on the Clip Similarity metric may result in a lower score for formations that better correspond to the given text for an average user.

- [10] J. Alonso-Mora, A. Breitenmoser, M. Rufli, R. Siegwart, and P. Beardley, “Multi-robot system for artistic pattern formation,” in *IEEE Int. Conf. on Robotics and Automation*, 2011, pp. 4512–4517.
- [11] J. Alonso-Mora, A. Breitenmoser, M. Rufli, R. Siegwart, and P. Beardley, “Image and animation display with multiple mobile robots,” *The International Journal of Robotics Research*, vol. 31, no. 6, pp. 753–773, 2012.

- [12] S. Hauri, J. Alonso-Mora, A. Breitenmoser, R. Siegwart, and P. Beardley, “Multi-robot formation control via a real-time drawing interface,” in *8th Int. Conf. of Field and Service Robotics*, 2013, pp. 175–189.
- [13] M. Waibel, B. Keays, and F. Augugliaro, “Drone shows: Creative potential and best practices,” ETH Zurich, Tech. Rep., 2017.
- [14] H.-J. Kim and H.-S. Ahn, “Realization of swarm formation flying and optimal trajectory generation for multi-drone performance show,” in *IEEE/SICE Int. Symposium on System Integration*, 2016, pp. 850–855.
- [15] H. Sun, J. Qi, C. Wu, and M. Wang, “Path planning for dense drone formation based on modified artificial potential fields,” in *2020 39th Chinese Control Conference*, 2020, pp. 4658–4664.
- [16] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, “Monte carlo localization for mobile robots,” in *IEEE Int. Conf. on Robotics and Automation*, vol. 2, 1999, pp. 1322–1328.
- [17] “Convex hull library,” <https://github.com/EtzionR/My-Convex-Hull>, accessed: 2023-03-2.
- [18] S. Shah, D. Dey, C. Lovett, and A. Kapoor, “Airsim: High-fidelity visual and physical simulation for autonomous vehicles,” in *Field and service robotics*, 2018, pp. 621–635.
- [19] P. Pueyo, E. Cristofalo, E. Montijano, and M. Schwager, “Cinemairsim: A camera-realistic robotics simulator for cinematographic purposes,” in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2020, pp. 1186–1191.
- [20] A. Koubâa et al., *Robot Operating System (ROS)*. Springer, 2017, vol. 1.