Revista de Informática Teórica e Aplicada - RITA - ISSN 2175-2745
Vol. 28, Num. 02 (2021) 39-49

RESEARCH ARTICLE

Centralized Algorithms Based on Clustering with Self-tuning of Parameters for Cooperative Target Observation

Algoritmos Centralizados baseados em Agrupamento com auto-ajuste de Parâmetros para Observação Cooperativa de Alvos

João Pedro B. Andrade[1]*, José Everardo B. Maia[1], Gustavo Augusto L. de Campos[1]

Abstract: Clustering on target positions is a class of centralized algorithms used to calculate the surveillance robots' displacements in the Cooperative Target Observation (CTO) problem. This work proposes and evaluates Fuzzy C-means (FCM) and Density-Based Spatial Clustering of Applications with Noise (DBSCAN) with K-means (DBSk) based self-tuning clustering centralized algorithms for the CTO problem and compares its performances with that of K-means. Two random motion patterns are adopted for the targets: in free space or on a grid. As a contribution, the work allows identifying ranges of problem configuration parameters in which each algorithm shows the highest average performance. As a first conclusion, in the challenging situation in which the relative speed of the targets is high, and the relative sensor range of the surveillance is low, for which the existing algorithms present a substantial drop in performance, the FCM algorithm proposed outperforms the others. Finally, the DBSk algorithm adapts very well in low execution frequency, showing promising results in this challenging situation.
Keywords: Multi-Agent Systems — Agent-Based Simulation — Clustering Methods — Intelligent Robots

Resumo: O agrupamento de posições de alvos é uma classe de algoritmos centralizados usados para calcular os deslocamentos de posições dos robôs de vigilância no problema de Observação Cooperativa de Alvos (CTO). Este trabalho propõe e avalia algoritmos centralizados de agrupamento com auto-ajuste de parâmetro baseados em Fuzzy C-means (FCM) e DBSCAN com K-means para o problema CTO e compara seu desempenho com o algoritmo K-means. Dois padrões de movimento aleatório são adotados para os alvos: no ambiente livre ou em uma grade de obstáculos. Como contribuição, o trabalho permite identificar faixas de parâmetros de configuração de problemas em que cada algoritmo apresenta melhor desempenho médio. Como uma primeira conclusão, na situação desafiadora em que a velocidade relativa dos alvos é alta e o alcance relativo do sensor da vigilância é baixo, para a qual os algoritmos existentes apresentam uma forte queda no desempenho, o algoritmo FCM proposto supera os outros. Por fim, o algoritmo DBSk se adapta muito bem em situações com baixa frequência de execução, apresentando bons resultados nesta situação desafiadora.
Palavras-Chave: Sistemas Multiagentes — Simulação Baseada em Agente — Métodos de Agrupamento — Robôs Inteligentes

[1] *Universidade Estadual do Ceará (UECE), Fortaleza - Ceará, Brasil*
***Corresponding author**: joaopbern7@gmail.com
DOI: http://dx.doi.org/10.22456/2175-2745.107154 • **Received:** 31/08/2020 • **Accepted:** 23/05/2021

1. Introduction

The Cooperative Target Observation (CTO) problem domain is one in which a team of moving surveillance robots, for example, drones, must observe another target robot team in motion to maximize the Average Number of Observed Targets (ANOT) in a period. Various practical problems, such as target recognition, security, and surveillance, are in the CTO problem domain. A typical scenario of this application can be seen in Figure 1.

In practice, the currently observing entities are Unmanned Aerial Vehicles (UAV) [1], Unmanned Ground Vehicles (UGV) [2], Unmanned Underwater Vehicles (UUV) [3]. Among others, some applications use micro-drones for performing surveillance [4], unmanned aerial systems, and GPS data recorders joined to conduct ecological research [5], and that performs population and social monitoring through UGVs and UAVs, among other situations, [6].

The field was initially explored in [7], where the Cooperative Multi-Robot Observation of Multiple Moving Targets (CMOMMT) was defined. This problem has two robot groups, observers and targets, in a 2-dimensional environment; observers have 360° of view observation sensors, where the mission is to keep targets under observation, both robot types can move. Cooperative Target Observation (CTO) is a variant CMOMMT introduced by [3]. The CTO problem domain has various instances depending on the type of movement of the targets, resource constraints, the interaction between targets and observers, and the stated specific objective.

The survey in [8] presents a classification of problems related to CTO. In the approach adopted in this paper, the team's objective is to maximize the observation of targets in a fully observable environment, in which the targets cooperate with the observers by informing their locations, so it minimizes the total time in which the targets escape of the observation of the observer's group. Observers work collectively to observe each target by at least one time, rather than an observer staying all observation time on a target.

In [9], it was proposed two ways of decision-making for observers, a K-means-based approach and a Hill-Climb-based approach, adjustable degree of centralization of the team reasoning, showing best results with centralized algorithms. Recently in [10], the decentralized K-means approach was extended with additional behaviors that increase observers' autonomy level. Although very efficient in some situations, the K-means algorithm has some problems when applied in complex configurations, i.e., observers with short range-sensor targets with high-speed and limited processing. The emergence of empty clusters during the clustering process with the K-means algorithm is another factor that degrades the observer performance.

To avoid these problems and improve the performance in complex CTO configurations, we propose two new algorithms, a Fuzzy C-means (FCM) [11] based and other approach exploring dense regions with many targets called Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [12] associate with K-means, called DBSk. Both algorithms depend on the right choice of input parameters to works as well as possible. We propose a generic function for tuning its input parameters during the experiment based on algorithm performance. Two random motion patterns were adopted for the targets: free space and a grid. The baseline results for our research are those in [9].

This paper is structured in three more sections. Section 2 defines the problem model and algorithms. Experiments and the results are shown in Section 3. Finally, Section 4 concludes the paper by describing the final remarks and future research.

2. Model and Algorithms

Figure 2 describes the CTO problem, as defined by [9]. The Observers and the targets are mobile agents that move in a continuous environment, non-toroidal rectangular 2-Dimensional field free of obstacles. Observers attempt to keep under observation as many targets as possible while the targets move randomly in the environment. Each observer can see targets around him through an R limited range radial sensor. Targets inform their locations, acting cooperatively to be observed by observers. The general operation in [9] is a cyclic procedure. The coordinator computes the new destination point and sends it to each observer every γ time step. If one reaches its destination point in less than γ time steps, it waits until a new destination point is computed.

The targets do not try to avoid the observers and move randomly throughout the space. The movement of the targets is done by setting a destination point, then having the travel agent towards this point. The targets travel towards their destination point at most 100 time-steps. If they reach the destination before 100 time-steps, they can then compute to a new destination point immediately. If it does not, they calculate a new destination and follow. Targets' destination points are chosen at random from within a local region (one-quarter of the environment height and width) centered on the target: the locality helps prevent targets from clustering near the center of the field on the way to their intended destinations.

In our approach, we are interested in ways to maximize measures of performance of observation. In [9], the best results were obtained with centralized solutions, where all observers were in the same group. The decision-making on the group is made by the coordinator that receives the targets' location and assigns destinations for each observers' member of the group. We model our solution in the same way, where a coordinator assigns destinations for the other members, and we apply new ways of calculating the destinations through FCM clustering and by a combination of DBSCAN with K-means (DBSk). The performances of these proposals are compared against that k-means based in [9].

The objective of the observers is the same objective defined by [7], which maximizes the average of observed targets during the experiment time. Here, similar to formalism presented in [13], at each time step t, the observers compute which targets are monitored, given by the indicator function $\Theta(\omega)$, where their value is 1 if ω is monitored at t and 0, otherwise. Where ω is any target, then at each time-step is possible to know the number of targets observed by the observers through a μ value computed by:

$$\mu(t) = \sum_{\omega \in \Omega} \theta(\omega) \tag{1}$$

in which Ω stands the conjunct of the all targets ω. The average value of μ considering all the experiment's time-steps is the function to be maximized by the organization as its objective, defined by:

$$\bar{\mu} = \frac{1}{T} \sum_{t \in T} \mu(t) \tag{2}$$

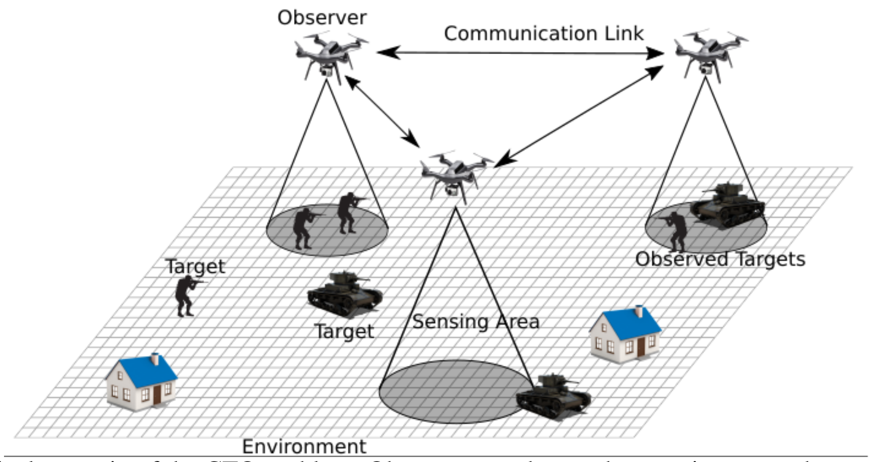


Figure 1. A typical scenario of the CTO problem. Observers are drones that monitor ground operations on targets.

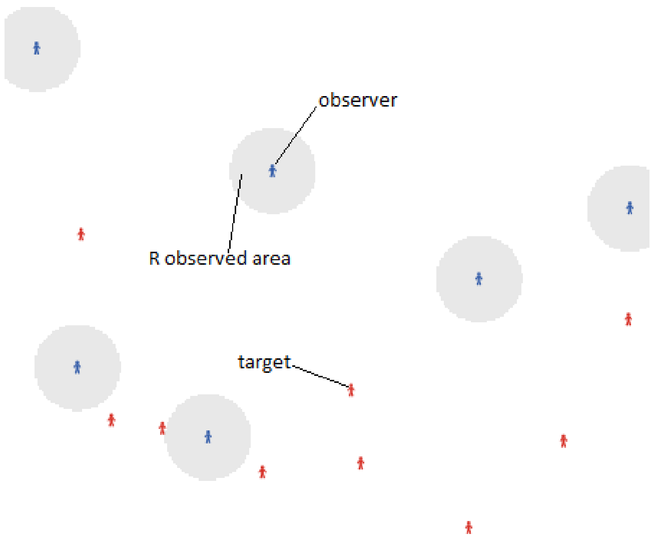


Figure 2. Screenshot of the model showing a configuration of targets and observers with their R sensor range.

where T stands the conjunct of the t time-steps, composing the total time of the experiment. We call the $\bar{\mu}$ value of Average Number of Observed Targets (ANOT), and this value will be considered the performance measure of the observers.

2.1 Fuzzy C-means

The K-means-based algorithm uses the observers' location as the initial cluster's center and the location of the targets as points to be clustered. It clusters data points by similarity measure (Euclidean distance), then the new destinations are calculated as the average of data points of each cluster, assigned the movement to each observer to a new point calculated. However, in the K-means cluster, the algorithm with initial centers assigned can result in empty clusters, as discussed in [14]. When this happens, the observer of the empty cluster has no destination to be assigned. Ad hoc solutions are then adopted to correct such cases.

In [9] and [10] when an empty cluster happens, is assigned a random point as the destination. However, another way to deal with the empty cluster problem can improve the performance in CTO, avoiding observer moving to random points. We propose to apply FCM clustering replacing k-means, where the empty cluster problem exists, with each point belonging to many clusters at the same time.

Proposed in [15] and improved by [11], FCM is a standard clustering algorithm that groups data points in a specific number of clusters. Given the input data values, we determine the set centers C_i and the membership matrix U, containing each data point's membership value concerning the determined centers. The algorithm is based on the minimization of the following objective function:

$$J_m = \sum_{i=1}^{N} \sum_{j=1}^{C} u_{ij}^m ||x_i - c_j|| \quad , \quad 1 \leq m < \infty \tag{3}$$

in which C is the set of all clusters, m is a number greater than one, that defines the allowed distance between data points and centers, u_{ij} is the degree of membership of x_i from the cluster j, where x_i is the i-th of n-dimensional measured data, c_j is the d-dimension center of the cluster, and $||*||$ is the similarity measure between any measured data and any center.

$$u_{ij} = \sum_{k=1}^{C} \left[\frac{||x_i - c_j||}{||x_i - c_k||} \right]^{\frac{-2}{m-1}} \quad , \quad c_j = \frac{\sum_{i=1}^{N} u_{ij}^m \cdot x_i}{\sum_{i=1}^{N} u_{ij}^m} \tag{4}$$

At every step of the iterative optimization of 3, the values u_{ij} and c_j are recalculated according to 4. This process is repeated until no more significant progress is made in 3.

To apply the FCM in CTO, similar to [7] with K-means, as proposed in [16] but without self-tuning of parameters, we consider the observers to be centers and the targets to be data points and observers' destinations as the centers of the clusters. The initial cluster centers are the current positions of the observers. There is a rate of algorithm update γ where the coordinator performs the algorithm with locations received of the targets and assigns each new center cluster to subordinate observers.

2.2 DBSCAN

DBSCAN, proposed in [12], is a clustering non-parametric method based on density, being significantly adequate to find clusters with arbitrary shape and size by counting points in regions with this size. The key idea is that, for each data point in a cluster, the neighborhood in a radius of this point has to be composed by a number given a minimum number of neighbor points. This way, the clusters are identified as dense regions, and any other points out of these regions are considered noise points. This algorithm's use arises from the insight that small regions with high target density should be observed and not divided into different clusters.

This algorithm requires two input parameters: the radius ε (define the ε-neighborhood of a p point) and $MinPts$, the minimum number of points in this ε-neighborhood to be considered a cluster. Each data point has a ε-neighborhood given by:

$$N_\varepsilon(p) = \{\forall q \in D | dist(p,q) < \varepsilon\} \tag{5}$$

where p and q belong to D, the conjunct with all data points clustered. When an area of interest is discovered, according to ε and $MinPts$, are defined the central point p and the other points that have fewer neighbors than $MinEps$ are the edge points. The cluster in DBSCAN is formed through points in a ε-neighborhood and density-reachable (with a chain respecting ε and $MinEps$). All the other points not clustered by DBSCAN are considered noise-points.

2.3 DBSk: Combining DBSCAN with k-means

In CTO, the points are targets, and the value is an approximated value of the observers' sensor-range, and the destiny point for observers is the centroid of each cluster. The different sensor-range values in CTO can result in many or few noise-targets; besides that, in so many cases, the cluster numbers found are less than observers, causing observers without destiny.

We propose to combine DBSCAN with k-means, using k-means to cluster noise-targets, where the observers' distance partitions the targets. We used ε value $= 1$, as described by [10] as the better configuration value. The DBSk algorithm proposed to resolve this problem and exploring dense regions work as follows:

1. Targets are clustered with DBSCAN.
2. Are calculated all center points of each DBSCAN-cluster, then assigned for the closest observer for each point.
3. If the number of targets-noise is greater than the number of observers without destiny, then target-noisy is clustered with K-means.
4. If observers are left without a destination, they are assigned the closest target as destiny for these observers.

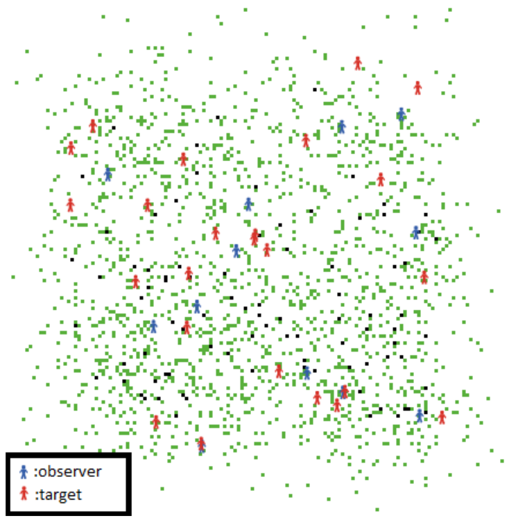
We studied the behavior of the DBSk algorithm. This is shown in Figure 3. Each subfigure is the memory of a complete run of the simulation. K-means calculated green points, black points were calculated by DBSCAN, and yellow points were calculated for another condition stated in step 4 from DBSk. Note that from left to right as the range sensor grows, the number of green dots decreases and yellow dots grow, and the number of black dots grows from Figure 3a to Figure 3b, and then decreases in Figure 3c. Thus, in scenarios with low values for sensor range (used as ε), few clusters are discovered, generating many noise-targets clustered by K-means. In cases with high sensors range, DBSCAN generates few noise-targets making the combination flexible to meet a wide range of sensor range. To follow the insight, we want the maximum black points. This will be done through the self-tuning of ε by the algorithm.

2.4 Self-tuning Function

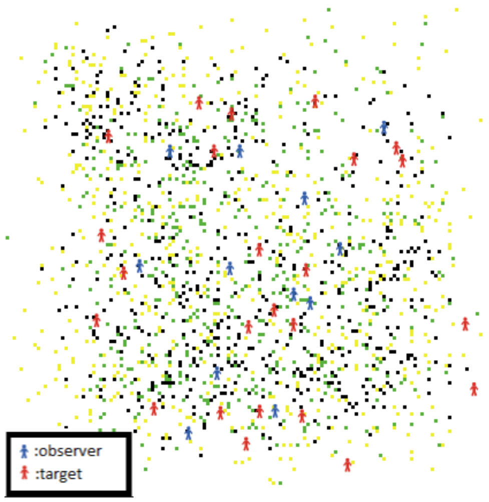
The performance of the FCM and the DBSk algorithms are sensitive to the input parameters. In FCM, the m value, as discussed in [17], is vitally important to the algorithm, where values close to inferior limit 1 tends to reduce the participation of the targets in many clusters, and for large values, all targets tend to belong to all clusters at the same time. In this situation, different configurations work better with distinct m values. Therefore, a local search auto-tuning function is used to tune these parameters at run time.

In DBSk, the step DBSCAN clustering is sensitive to the ε value according to the configuration, where for large range-sensors, lower ε values can avoid large unit clusters, and for small range-sensors, values greater can make sure DBSCAN clusters. High speed-targets can undo clusters with small ε values. Fixed values for m and ε can decrease ANOT for different configurations; then, we propose a method for online self-tuning of the value based on the previous values' performance.

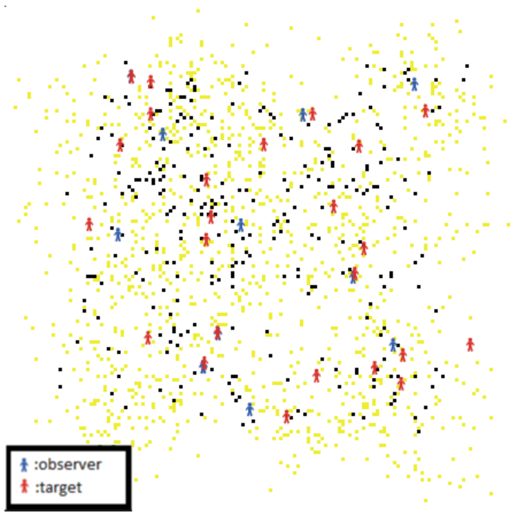
As seen in Algorithm 1, ρ is the parameter value to be tuning, ϕ is an arbitrary value that defines the absolute displacement value of the parameter, signal defines the direction of displacement ($+1$ or -1), and finally, Δ is ANOT value measured through the γ steps since that last time the algorithm performed, and i index of Δ is the current time-step where which is performed the algorithm. Every time the performance decreases, the algorithm inverts the displacement value searching for the best value for a given CTO configuration. In the Algorithm 1, the signal input is required to enable self-tuning in the case where the parameter is of the smaller-better type or when it is of the higher-better type.



(a) Range-sensor and $\varepsilon = 5$



(b) Range-sensor and $\varepsilon = 15$



(c) Range-sensor and $\varepsilon = 25$

Figure 3. Final screenshot of runs whit DBSk destination points calculated in a experiment with $\gamma = 10$ and ε fixed as range-sensor for 1500 time-steps

Algorithm 1: Self-Tuning of Parameters function

```
   Data: $signal$, $\rho$, $\phi$ , $\Delta_i$, $\Delta_{i-1}$ (the two last ANOT
         values calculated)
   Result: $\rho_{new}$ (the new parameter value)
1  $Initialization$ : $signal \leftarrow 1$ if signal is null
2  if $\Delta_i - \Delta_{i-1} < 0$ then
3      $signal \leftarrow signal * -1$
4  end
5  $\rho_{new} \leftarrow \rho + (\phi * signal)$
6  return $\rho_{new}$
```

3. Results

In this section, we present all the experimental results of this work, starting by the study the effect of the γ update rate on the performance of the algorithms in Section 3.1. We study in Section 3.2 how the parameters m and ε influence the performance of the proposed algorithms, thus justifying the need for the self-fitting function. Also, we show the comparative performance results between FCM, DBSk, and K-means. In Section 3.3, the movement of targets occurs in an open environment within the boundary. In Section 3.4, the movement of targets is restricted to occur on a rectangular grid.

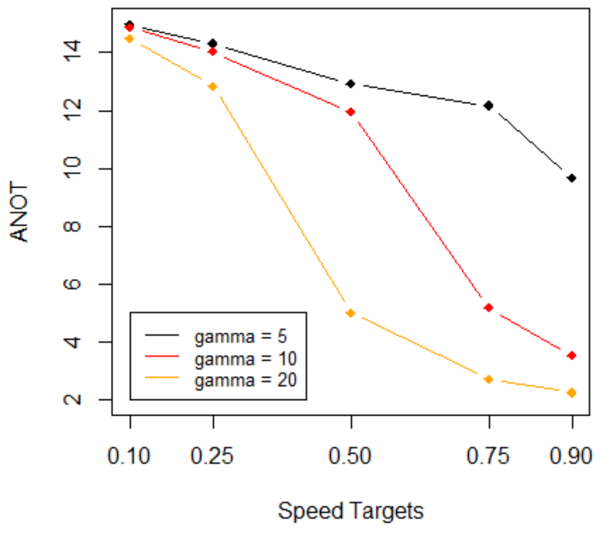
3.1 Update Rate

As in [9], we tested the impact of the γ value in our approaches to know how resistant these algorithms are to increase the γ value; it may symbolize communication and processing difficulties. In one case, we picked γ value from $\{5, 10, 20\}$, targets speed value in 0.9, varying the sensor range from $\{5, 10, 15, 20, 25\}$; in the other case, varying targets speed for $\{0.1, 0.25, 0.5, 0.75, 0.9\}$ with range-sensor in 5.These cases are the harder scenario possible in our simulations for observers. The results are shown in Figs. 4 and 5.

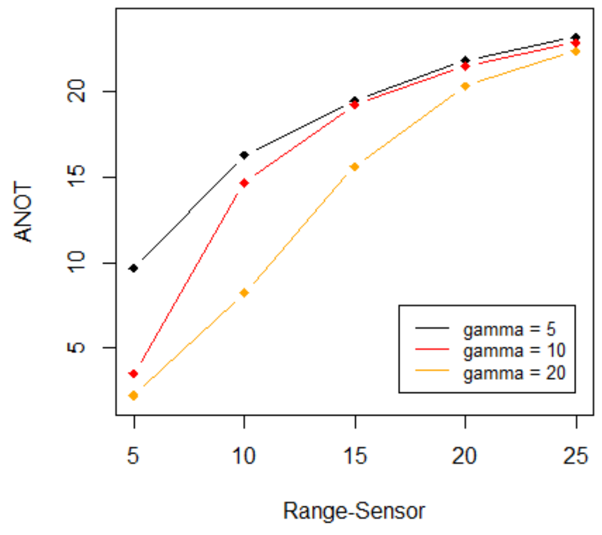
As expected, ANOT decreases with increasing targets' speed for all algorithms with the fixed sensor range in Figure 4 and increases with increasing sensor range with fixed targets speed in Figure 5. However, comparing the respective performance, DBSk showed better resistance to increase to γ values, where ANOT lines are closer together, Figures 5b and 4b, than in FCM case, Figures 4a and 5a. The dense regions explored in the DBSk take more time to dismount.

3.2 Paramerters m and ε

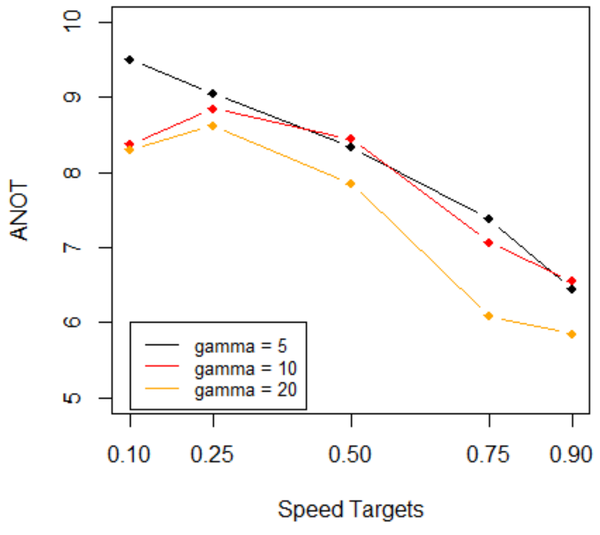
In order to investigate the ranges of active values of m and ε of the algorithms proposed in different configurations, we apply tests with all possible values for range-sensor and speed-targets and using $\gamma = 10$. We limited the values of the m for $\{m > 1$ and $m < 3\}$ with an initial value as 2 and $\phi = 0.05$. For ε we limited the value between 2.5 and 25, with 15 as the initial value and $\phi = 0.5$ and $MinPts = 2$. We perform the mean value of the final values of m and ε through 30 repetitions. These configurations were used in all other experiments. The results are shown in Tables 1 and 2.



(a) FCM varying speed-targets

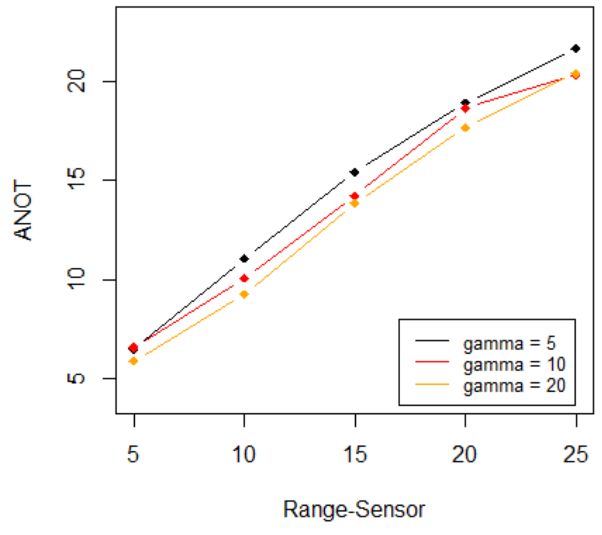


(a) FCM varying range-sensor



(b) DBSk varying speed-target

Figure 4. Comparison ANOT for different γ values varying Speed-Targets (sensor range = 5)



(b) DBSk varying range-sensor

Figure 5. ANOT for different γ values varying Sensors-Range (targets-speed = 0.9)

Table 1. The best m values found in $\gamma = 10$ (FCM)

Sensor	Targets Speed				
Range	*0.1*	*0.25*	*0.5*	*0.75*	*0.9*
5	2.395	2.495	2.475	2.230	2.145
10	2.070	1.695	2.030	2.060	2.460
15	1.655	1.690	2.105	2.410	2.060
20	1.520	2.160	1.990	2.445	2.320
25	1.990	1.925	1.780	2.165	2.260

Table 2. The best ε values found in $\gamma = 10$ (DBSk)

Sensor	Targets Speed				
Range	*0.1*	*0.25*	*0.5*	*0.75*	*0.9*
5	18.50	19.25	18.80	19.85	19.95
10	13.80	13.20	18.55	16.05	15.20
15	11.70	14.60	17.10	13.60	18.10
20	15.55	12.35	13.65	17.10	14.80
25	12.25	14.20	14.70	18.50	18.00

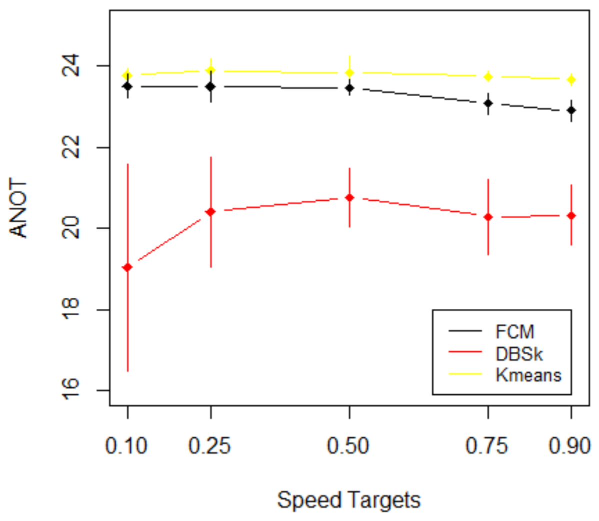


Figure 6. ANOT for Sensor-Range = 25 and $\gamma = 10$.

In Table 1, we noted that for small values for targets-speed and large values for sensors-range, the m value was decreased and increased for short sensor-range and high speed-targets that significant fuzzy groups outcome in best results when the configuration is complicated (small sensor-range and high target-speed).

In Table 2, small values in the sensor range result in high best values for ε showing an inversely proportional growth, and except for the sensor range = 20, ε is directly proportional to targets-speed. In short sensor-range, the increase of the ε was to decrease the targets-speed, exploring dense regions, and too large range-sensors high ε values can result in few clusters, then this value is naturally decreased for make clusters with K-means and target-nearest (step 4 in DBSk).

Tables 1 and 2 show the importance of adopting self-tuning. Note that the optimal values of m parameter vary between 1.520 and 2.495, and the ε parameter varies between 11.70 and 19.25, which are variations of approximately 60%.

3.3 Experiments in an Unrestricted Environment

To compare the algorithms, we use K-means [9] with α value = 1, the best configuration possible, as stated by [10] and we found the different situations where each algorithm is best than others.

Figure 6 is conclusive as to the superiority of k-means when the relative sensor range is large. K-means has the added advantage of being light processing and allowing larger γ values. On the other hand, when the relative sensor range is small, Figure 7 shows that FCM outperforms the others for all relative target speeds. However, note that FCM requires higher processing and higher update rate (lower γ), which implies higher communication costs.

FCM was best in cases with $\gamma = 5$ and 10, for small range-sensor and not high-speed-targets, as seen in Figure 8 the ANOT is considerably larger than other algorithms. The fuzzy clusters from FCM have centers closer to targets because of the pertinence calculated, and other case is the non-existence of empty clusters in FCM.

DBSk is more resistant to the increase of the γ value, as seen in the previous subsection, and the best performance of the algorithm was in high γ values, and high-speeds and low range-sensor. In Figure 9, the ANOT value for DBSk is better than others, and in Figure 8, where the exploration of dense regions with DBSCAN outcome good results.

The hypothesis of the Gaussian distribution of the results of 30 executions was adopted to perform the t-student and chi-square hypothesis tests for all the comparisons in Figures 6-9, which directly implicate the conclusions. Two-tailed tests of significance were performed with 95% confidence. For all comparisons, a P-value of < 0.05 was considered statistically significant.

3.4 Experiments in a Grid Environment

We also experimented in a scenario in which targets have restricted movements on a grid while observers move freely. We use a Manhattan grid as the first approximation of an urban environment, where the movement of the vehicles is restricted to the streets. In CTO-grid, the blocks are uniform rectangles 13×13 units equally separated by 8 units (width of streets) units one to the other, arranged as a grid and was developed as an agent-based simulation with NetLogo 6.0.4[1], with integration with software R[2] and can be seen in Figure 10.

As seen in Figure 10, the border of the street in north and east has 4 units complete, totalizing 49 obstacles (gray color),

[1]https://ccl.northwestern.edu/netlogo/
[2]https://www.r-project.org/

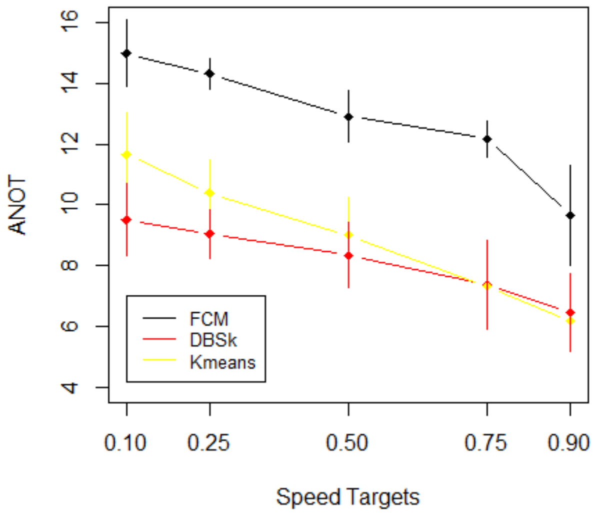


Figure 7. ANOT for Sensor-Range $= 5$ and $\gamma = 5$.

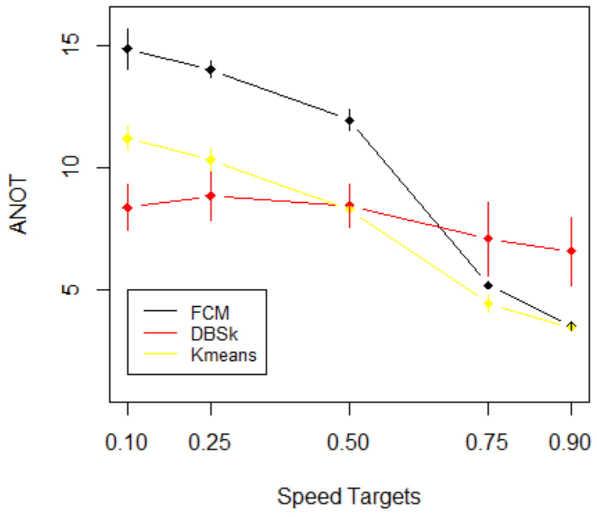


Figure 8. ANOT for Sensor-Range $= 5$ and $\gamma = 10$.

the red cars are the bus targets, and blue airplanes are the UAVs observers. In the experiments, the algorithms' performances based on K-means, FCM, and DBSk were compared. To measure the algorithms' performance, we do the sort of the target random destinations similar to [9] with a restriction of this point is a point in the street, avoiding obstacles Algorithm 2.

Algorithm 2: Target Behavior in the Grid Environment

```
if not exists a goal position OR distance to goal position is < speed-targets then
    calculate a random new point as the goal where that position has a maximum distance of < 25% of the total size of the environment if the goal is part of an obstacle then
        rerun this algorithm from the beginning
    end
    Set the new point calculated as the goal and set the orientation for this point
end
if there an obstacle ahead then
    Set a new orientation between north, south, east, or west with less angular displacement
end
Forward to goal point
```

We compared K-means with $\alpha = 1$, with FCM and DBSk in all configurations possible. Initially, the K-means consumes considerably less computational time, however in challenging situations (small range-sensors, high-speed targets, and high γ values), FCM and DBSk compensated the high computational time with good ANOT values.

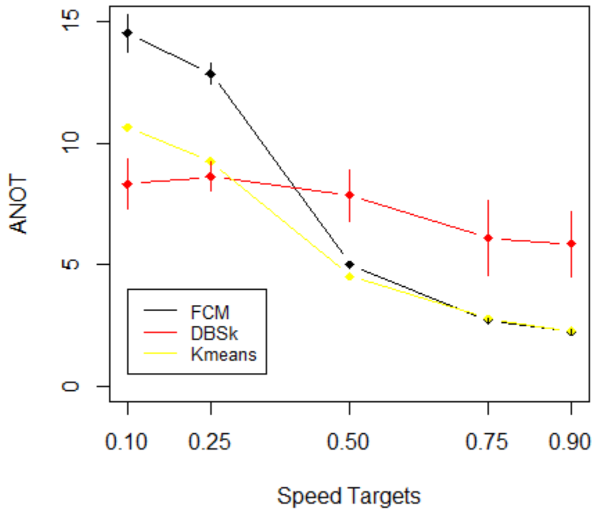


Figure 9. ANOT for Sensor-Range $= 5$ and $\gamma = 20$.

Table 3. The best m values found in $\gamma = 10$ (FCM) for grid environment

Sensor	Targets Speed				
Range	*0.1*	*0.25*	*0.5*	*0.75*	*0.9*
5	2.500	2.460	2.595	2.190	2.075
10	2.155	1.985	1.915	2.215	2.150
15	1.800	1.670	2.040	2.190	2.100
20	1.960	2.130	1.945	2.180	2.135
25	1.895	2.160	2.180	2.135	2.180

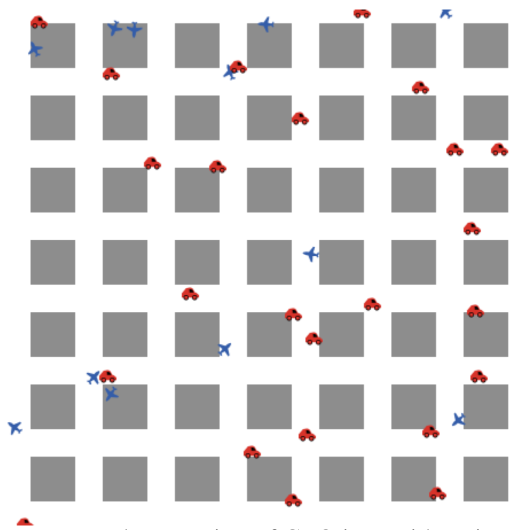


Figure 10. Implementation of CTO in a grid environment

Table 4. The best m values found in $\gamma = 10$ (DBSk) for grid environment

Sensor Range	Targets Speed *0.1*	*0.25*	*0.5*	*0.75*	*0.9*
5	14.90	16.40	19.05	15.80	18.80
10	14.55	15.80	15.05	16.55	16.05
15	16.10	15.60	17.15	19.35	15.40
20	13.85	17.50	13.15	12.55	14.95
25	9.60	15.50	16.55	14.20	19.50

Tables 3 and 4 show that the optimal values of the algorithms' parameters do not suffer the effect of the restriction on the movement of the targets. Coherently, Figures 11, 12 and 13 show that the positions in the performance rank of the algorithms also do not change. Figure 14 plus one confirms that for Sensor Range, large K-means is the recommended algorithm.

The hypothesis of the Gaussian distribution of the results of 30 executions was adopted to perform the t-student and chi-square hypothesis tests for all the comparisons in Figures 11-14, which directly implicate the conclusions. Two-tailed tests of significance were performed with 95% confidence. For all comparisons, a P-value of < 0.05 was considered statistically significant.

4. Conclusions and Future Works

In this paper, we propose two new centralized approaches to improve observers' performance in the CTO problem in a difficult situation, where the targets move at high relative velocity, and the observer sensor is of low range. Since these algorithms are parameter-dependent, we propose a function to adapt the algorithms or different configurations online, making them self-tuning. The proposed algorithms, FCM and DBSk, were analyzed, and their performances compared to k-means proposed in the literature.

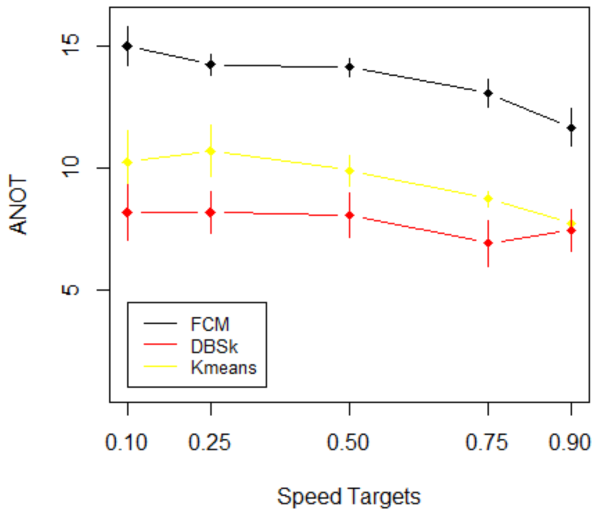


Figure 11. Performance of different speed targets with range-sensor = 5 and $\gamma = 5$ for the grid environment

From this research, some conclusions can be stated. When in suitable configurations, in which the relative speed of the targets is low and the close range of the sensors is high, the algorithm based on k-means is superior. At the other extreme, when the sensor range is low, and the relative speed of the targets is high, the DBSk algorithm outperforms the others. Finally, when the relative velocity of the targets is high, the sensor range is also low, the algorithm based on FCM is recommended; this is because the algorithm based on k-means suffers from the empty cluster problem in this configuration, which FCM avoids.

The relative performance of the algorithms was not affected when the movement of the targets is restricted to a grid environment, and it should be noted that the higher computational cost of the FCM and DBSk is compensated by a lower execution frequency required, as shown in the presented experiments (larger γ). It was not evaluated in this work, but during its development, it was evidenced that the dynamic self-tuning of the algorithms can also be used to deal with the local minimum problems commonly presented by K-means clustering algorithms. This will be investigated in another work.

Author contributions

- João Andrade performed the research, drafted and implemented the algorithms and simulation, conducted the experiments, collected the results, performed the statistical testing, and wrote the manuscript.
- Gustavo Campos was the primary research advisor who designed the entire paper, including the grid environ-

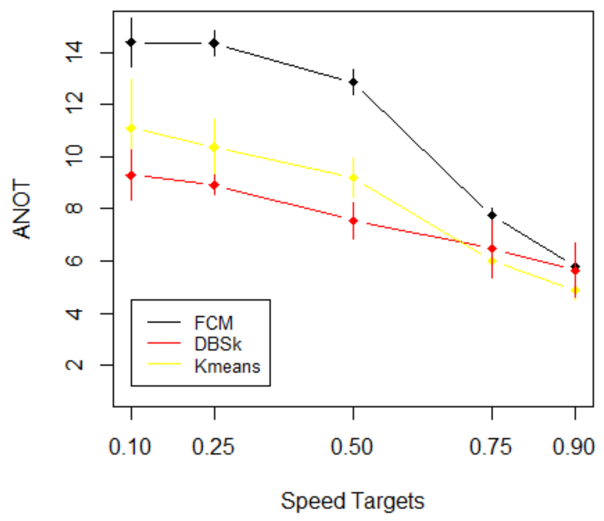


Figure 12. Performance of different speed targets with range-sensor = 5 and $\gamma = 10$ for the grid environment

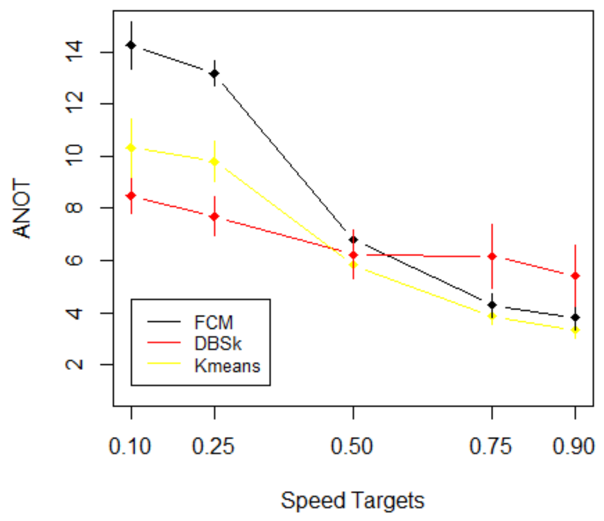


Figure 13. Performance of different speed targets with range-sensor = 5 and $\gamma = 20$ for grid the environment

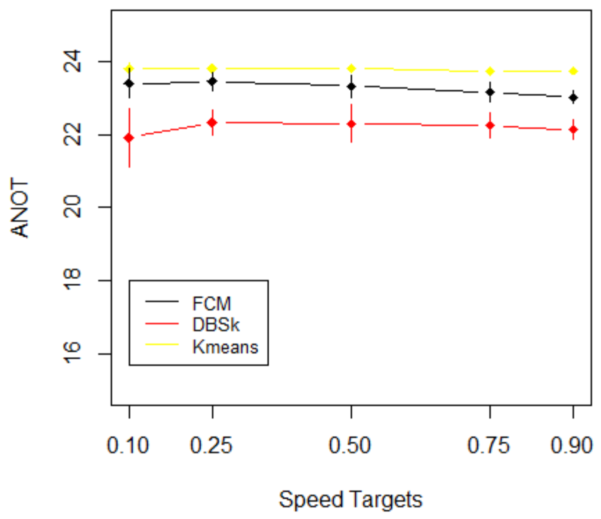


Figure 14. Performance of different speed targets with range-sensor = 25 and $\gamma = 10$ for grid the environment

ment, helping with writing the manuscript and design of the experiments.

- José Maia was the secondary research advisor who mainly contributed to the design of the clustering algorithms and the statistical test, and the writing of the paper.

References

[1] KANISTRAS, K. et al. A survey of unmanned aerial vehicles (uavs) for traffic monitoring. In: IEEE. **2013 International Conference on Unmanned Aircraft Systems (ICUAS)**. Atlanta, 2013. p. 221–234. Disponível em: ⟨https://doi.org/10.1109/ICUAS.2013.6564694⟩.

[2] BONADIES, S.; LEFCOURT, A.; GADSDEN, S. A. A survey of unmanned ground vehicles with applications to agricultural and environmental sensing. In: INTERNATIONAL SOCIETY FOR OPTICS AND PHOTONICS. **Autonomous air and ground sensing systems for agricultural optimization and phenotyping**. Baltimore, 2016. v. 9866, p. 98660Q. Disponível em: ⟨https://doi.org/10.1117/12.2224248⟩.

[3] SULLIVAN, K. M.; LUKE, S. Autonomous uuv control via tunably decentralized algorithms. In: IEEE. **2004 IEEE/OES Autonomous Underwater Vehicles (IEEE Cat. No. 04CH37578)**. Sebasco, 2004. p. 47–53. Disponível em: ⟨https://doi.org/10.1109/AUV.2004.1431192⟩.

[4] FIORANELLI, F. et al. Classification of loaded/unloaded micro-drones using multistatic radar. **Electronics Letters**,

IET, Stevenage, v. 51, n. 22, p. 1813–1815, 2015. Disponível em: ⟨https://doi.org/10.1049/el.2015.3038⟩.

[5] RODRÍGUEZ, A. et al. The eye in the sky: Combined use of unmanned aerial systems and gps data loggers for ecological research and conservation of small birds. **PLoS One**, Public Library of Science, São Francisco, v. 7, n. 12, p. e50336, 2012. Disponível em: ⟨https://doi.org/10.1371/journal.pone.0050336⟩.

[6] KHALEGHI, A. M. et al. A DDDAMS-based planning and control framework for surveillance and crowd control via UAVs and UGVs. **Expert Systems with Applications**, Elsevier, Amsterdã, v. 40, n. 18, p. 7168–7183, 2013. Disponível em: ⟨https://doi.org/10.1016/j.eswa.2013.07.039⟩.

[7] PARKER, L. E. Cooperative robotics for multi-target observation. **Intelligent Automation & Soft Computing**, Taylor & Francis, Abingdon-on-Thames, v. 5, n. 1, p. 5–19, 1999. Disponível em: ⟨https://doi.org/10.1080/10798587.1999.10750747⟩.

[8] KHAN, A.; RINNER, B.; CAVALLARO, A. Cooperative robots to observe moving targets. **IEEE transactions on cybernetics**, IEEE, Piscataway, v. 48, n. 1, p. 187–198, 2016. Disponível em: ⟨https://doi.org/10.1109/TCYB.2016.2628161⟩.

[9] LUKE, S. et al. Tunably decentralized algorithms for cooperative target observation. In: **Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems**. Nova Iorque: [s.n.], 2005. p. 911–917. Disponível em: ⟨https://doi.org/10.1145/1082473.1082611⟩.

[10] ASWANI, R.; MUNNANGI, S. K.; PARUCHURI, P. Improving surveillance using cooperative target observation. In: **Proceedings of the 31st AAAI Conference on Artificial Intelligence**. São Francisco: AAAI Publications, 2017. Disponível em: ⟨https://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14882⟩.

[11] BEZDEK, J. C. **Pattern recognition with fuzzy objective function algorithms**. Berlim: Springer Science & Business Media, 2013.

[12] ESTER, M. et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In: **KDD'96: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining**. Menlo Park: AAAI Press, 1996. v. 96, n. 34, p. 226–231. Disponível em: ⟨https://dl.acm.org/doi/10.5555/3001460.3001507⟩.

[13] BANFI, J. et al. An integer linear programming model for fair multitarget tracking in cooperative multirobot systems. **Autonomous Robots**, Springer, Berlim, v. 43, n. 3, p. 665–680, 2019. Disponível em: ⟨https://doi.org/10.1007/s10514-018-9735-4⟩.

[14] PAKHIRA, M. K. A modified k-means algorithm to avoid empty clusters. **International Journal of Recent Trends in Engineering**, Academy Publisher, Barpeta, v. 1, n. 1, p. 220, 2009.

[15] DUNN, J. C. A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. **Journal of Cybernetics**, Taylor & Francis, Abingdon-on-Thames, v. 3, n. 3, p. 32–57, 1973. Disponível em: ⟨https://doi.org/10.1080/01969727308546046⟩.

[16] ANDRADE, J. P. B. et al. Organization/fuzzy approach to the cto problem. In: **2018 7th Brazilian Conference on Intelligent Systems (BRACIS)**. São Paulo: [s.n.], 2018. p. 444–449. Disponível em: ⟨https://ieeexplore.ieee.org/document/8575654/⟩.

[17] TORRA, V. On the selection of m for fuzzy c-means. In: ATLANTIS PRESS. **2015 Conference of the International Fuzzy Systems Association and the European Society for Fuzzy Logic and Technology (IFSA-EUSFLAT-15)**. Gijón, 2015. Disponível em: ⟨https://doi.org/10.2991/ifsa-eusflat-15.2015.224⟩.