ISSN 2614-0047

Dragonfly Algorithm for Crowd NPC Movement Simulation in Metaverse

Ong, Hansel Santoso [a,1], Hartarto Junaedi [a,2], Joan Santoso [a,3*]

[a] Institut Sains dan Teknologi Terpadu Surabaya, Surabaya Indonesia
[1] hansel@istts.ac.id; [2] hartarto@istts.ac.id; [3] joan@istts.ac.id
* corresponding author

ARTICLE INFO

Article history
Received December 18, 2021
Revised January 18, 2022
Accepted February 3, 2022

Keywords
Metaverse
Crowd Movement Intelligent
Metaheuristic
Dragonfly Algorithm
Particle Swarm Optimization

ABSTRACT

During The Pandemic Period, The Development Of Virtual Reality (Vr) In The Field Of Social Media (Metaverse) Is Very Fast To Give New Experiences. To Provide A New Experience, The Development Of A Supporting Virtual World As A Gathering Place Is Needed, To Support The Presence Of Others That Become A Factor Of Social Virtual Presence (Svr) Npc Is Required. Npc Crowds Will Be Tested In a Job Fair Case Study By Compared Dragonfly And Particle Swarm Optimization Algorithms. Algorithm Testing Will Be Adjustable With The Same Parameters And Profiles For Individuals And Objectives. After Experiment And Evaluation, Dragonfly Algorithms Was More Optimal And Provided Better Svr.

1. Introduction

The pandemic has limited movement and interaction between people, causing new communication challenges. The current development of Virtual Reality (VR) technology enables its users to interact with other people in the virtual world through avatars, known as Social Virtual Reality (SVR) [1]. As the virtual world is built, it is necessary to add supporters such as a crowd that can walk around the world to make it more similar to the real world. The crowd needs to be given artificial intelligence not to look stiff or move based on the pattern provided, according to [2], [3].

The movement of humans in the real world always avoids objects in front of them and pays attention to their surroundings. Dragonfly Algorithm (DA) by [4] is a metaheuristic algorithm that has the characteristic of knowing who its neighbors are and can walk side by side to the destination without bumping into each other and away from enemies, which is a modification of Particle Swarm Optimization (PSO) [5]. In the real world, it would more or less apply the same thing to not bumping into each other and getting past obstacles, and not crashing into walls. Implementing artificial intelligence for crowds can be done by simulating crowds in an exhibition. A job fair exhibition is a form of exhibition that can provide multiple profiles for each individual, and each has their abilities.

Previous studies found that many crowd simulations were driven by metaheuristic algorithms such as PSO and other modifications. However, DA for motion simulation still does not exist. Therefore the gap in this study is to use DA as an artificial intelligence for crowd simulation. In addition, the use of crowd simulations with many profiles in an exhibition is still minimal.

Through this research, it is hoped that DA will be a solution for crowd simulation of Non-Playable Character (NPC), which has more natural movements than other algorithms. In addition, through this research, it will be developed in a metaverse world that prioritizes the virtual world like the real world with VR [6].

2. Method

NPC crowd simulation in this study will compare PSO with DA with the same scenario and fitness function. The manufacturing stage will be divided into three parts: preparation, trial, and evaluation.

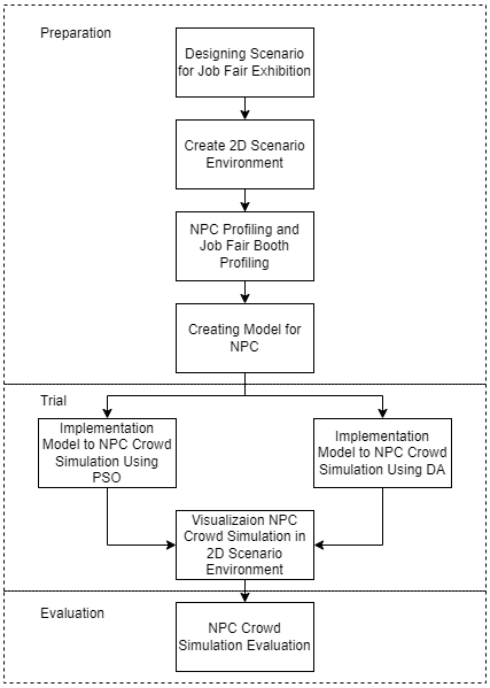

Fig. 1. The methodology is divided into three stages: preparation, trial, and evaluation.

The preparation stage will be divided into several parts, as in "Fig. 1", designing scenarios for job fair exhibitions, creating 2d scenario environments, NPC profiling and job fair booth profiling, and creating models for NPC. The first part will design the form of the Job Fair, from how wide the world will be made, how many booths will be used in 1 job fair, what is the minimum number of visitors and the maximum number of visitors, how many types of jobs will be adapted to the booth, and how many types of individuals.

2.1. Preparation

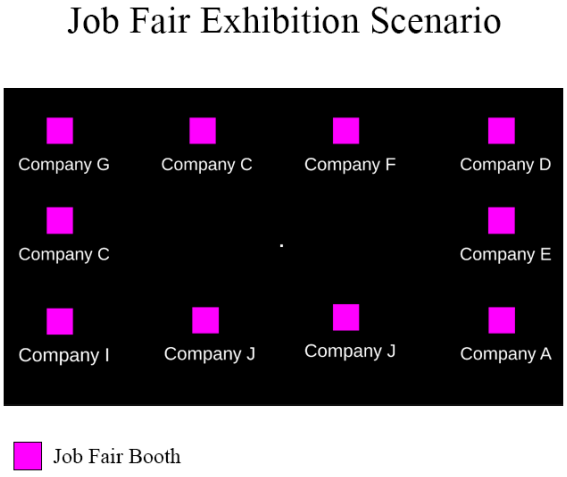

Fig. 2. Initial scenario design of job fair exhibition

The second part, like "Fig. 2" continuing with the scenario design is urgently needed to determine the position of the spawn location for each NPC, the position of the fair job booth and which places NPCs may enter or not, and exits from exhibitions such as job fairs in general so that they can

determine the flow of visitors. The third section will profile each type and booth type. The profile of each NPC can influence decision-making and how long it takes to reach the fair job booth.

Table.1 NPC profiling based on the skill

No.	Job Seeker	NPC Profiling				
		Skill	***Age***	***Degree***	***Interest***	***Score***
1	Person A	Artificial Inteligent	Random (23-40)	Bachelor	IT	0.0 - 0.1
2	Person B	Machine Learning Engineer	Random (23-40)	Bachelor / Master	IT	0.1 - 0.2
3	Person C	Data Scientist	Random (23-40)	Bachelor / Master	IT	0.2 - 0.3
4	Person D	Web Developer	Random (23-40)	Bachelor	IT	0.3 - 0.4
5	Person E	Accountant	Random (23-40)	Bachelor	Business	0.4 - 0.5
6	Person F	Business Analyst	Random (23-40)	Bachelor / Master	Business	0.5 - 0.6
7	Person G	Financial Analyst	Random (23-40)	Bachelor / Master	Business	0.6- 0.7
8	Person H	Animators	Random (23-40)	Bachelor / Master	Design	0.7- 0.8
9	Person I	Creative Director	Random (23-40)	Bachelor / Master	Design	0.8- 0.9
10	Person J	Fashion Designer	Random (23-40)	Bachelor	Design	0.9 - 1.0

NPC profiles like "Table 1" each profile of the NPC is divided into ten types according to the individual skills of each NPC. Other attributes that support the profile of the NPC are age which will be given a random score from the age of 23-40, graduates will also be randomly assigned between bachelor and masters, and interest will have a fixed value according to the skills of each NPC.

Table.2 Job Fair Booth Profiling Based on The Vacancy

No	Company	Job Fair Booth Profiling							
		Vacancy 1	***Vacancy 2***	***Vacancy 3***	***Minimum Requirement Degree***	***Min Age***	***Max Age***	***Field***	***Score***
1	Company A	Machine Learning Engineer	Data Scientist	Web Developer	Bachelor	23	26	IT	0.0 - 0.1
2	Company B	Creative Director	Artificial Inteligent	Business Analyst	Bachelor	23	40	Mixed	0.1- 0.2
3	Company C	Accountant	Financial Analyst	Data Scientist	Bachelor	23	40	Mixed	0.2 - 0.3
4	Company D	Artificial Inteligent	Machine Learning Engineer	Data Scientist	Master	24	30	IT	0.3 - 0.4
5	Company E	Business Analyst	Financial Analyst	Accountant	Bachelor	23	26	Business	0.4 - 0.5
6	Company F	Data Scientist	Business Analyst	Creative Director	Master	24	40	Mixed	0.5 - 0.6
7	Company G	Artificial Inteligent	Animators	Web Developer	Bachelor	23	26	Mixed	0.6- 0.7
8	Company H	Financial Analyst	Fashion Designer	Animators	Bachelor	23	30	Mixed	0.7 - 0.8
9	Company I	Fashion Designer	Creative Director	Animators	Bachelor	23	25	Design	0.8 - 0.9
10	Company J	Animators	Artificial Inteligent	Business Analyst	Bachelor	23	30	Design	0.9 - 1.0

The booth will also be given a profile that suits the needs of the NPC to make the NPC move toward the booth. Booth profiles are differentiated and given variations such as "Table 2" so that each NPC can move naturally and not based on existing patterns. The model prepared in the fourth section is intended for the fitness function that PSO and DA will use.

2.2. Trial

The trial phase will implement the model as a fitness function for PSO and DA, giving the same scenario conditions, such as the number of NPCs, booths, and exit and entry directions. In this stage, it is tested to see the movement of NPCs with different algorithms used, how long it takes for each NPC to reach its destination, and how long it takes to reach that goal. PSO developed and developed a coefficient to control velocity.

The following is a formulation that describes the position and velocity of an individual in a particular dimension and space in (1) and (2).

$$X_i(t) = X_{i1}(t) + X_{i2}(t), \ldots., X_{iN}(t) \quad (1)$$

$$V_i(t) = V_{i1}(t) + V_{i2}(t), \ldots., V_{iN}(t) \quad (2)$$

where,

X: Particle Postition

V: Particle Velocity

i: Particle Index

t: t-th iteration

N: space dimensions

The following is a model that describes the mechanism of movement of individuals [5]

$$V_i(t) = V_i(t-1) + c_1 r_1 (Pbest\ -\ X_i(t-1) + c_2 r_2 (Gbest\ -\ X_i(t-1) \quad (3)$$

Where,

$V_i(t)$: Velocity of each individual in each iteration

c_1, c_2: Learning factor.

r_1, r_2: Random number 0-1.

Pbest: The best position of the individual.

Guest: The best position of the population.

Following are some of the properties of PSO which are also found in DA and some of which have been modified to be calculated by the model [7]:

- Separation

$$S_i = -\sum_{j=1}^{N} X - X_j \quad (4)$$

Where X is the individual's current position, while X_j is the position of the j-neighbor. The number of neighbors is indicated by N [7].

- Alignment

$$A_i = \frac{\sum_{j=1}^{N} V_j}{N} \quad (5)$$

Where, V_j denotes the speed of the j-th individual neighbors, and N is the number of neighbors [7].

- Cohesion

$$C_i = \frac{\sum_{j=1}^{N} X_j}{N} - X \quad (6)$$

Where X denotes the individual's current position and X_j denotes the position of the j-neighbor individual, and N is the number of neighbors [7].

- Attraction toward a food source

$$F_i = X^+ - X \quad (7)$$

Where X indicates the individual's current position and X^+ indicates the position of the food source [7].

- Distraction outward an enemy

$$E_i = X^- + X \quad (8)$$

Where X denotes the individual's current position and X^- denotes the enemy's position [7].

To improve the position of each individual in exploring and exploiting, DA uses a modification of the PSO step vector as follows:

$$\Delta X_{t+1} = (sS_i + aA_i + cC_i + fF_i + eE_i) + w\Delta X_t \quad (9)$$

where:

ΔX: Vector step

s: separation weight

a: alignment weight

c: cohesion weight

f: food weight

e: enemy weight

w: inertia weight

The truncated step vector will be used for position vector calculations as follows:

$$X_{t+1} = X_t + \Delta X_{t+1} \quad (10)$$

To maximize exploration and exploitation with DA. DA uses Lévy flight calculations [8] when individuals have no neighbors. The following is the calculation of the position vector from the Lévy flight:

$$X_{t+1} = X_t + Lévy\,(d)\,X_{t+1} \quad (11)$$

The formula calculates Lévy flight:

$$Lévy\,(x) = 0.01\frac{r_1\sigma}{|r_2|^{\frac{1}{\beta}}} \quad (12)$$

where r_1, r_2 are random numbers 0-1, the β constant is 1.5, and σ is calculated by:

$$\sigma = \left(\frac{\tau(1+\beta)sin(\frac{\pi\beta}{2})}{\tau(\frac{1+\beta}{2})\beta\cdot 2^{(\frac{\beta-1}{2})}}\right)^{\frac{1}{\beta}} \quad (13)$$

To calculate τ by:

$$\tau(x) = (x-1)! \tag{14}$$

The calculation for each iteration of PSO and GA also has differences at each step. When PSO runs the fitness function for all individuals, the best individual (Gbest) will be selected, and the best movement will be recorded for each individual (Pbest). Meanwhile, DA first calculates each neighbor with the Euclidian distance and updates each individual's neighbors. When an individual alone does not have neighbors, the individual will move randomly to explore, and vice versa. When there is at least one neighbor, the individual will expand the radius of their neighbors to be more convergent.

The fitness function formulation to be used in this study is as follows:

$$\mathrm{d} = \sqrt{(x_2 - x_1)^2 - (y_2 - y_1)^2} \tag{15}$$

$$p = |x_2 - x_1| \tag{16}$$

$$f(x) = d + p \tag{17}$$

Equation (15) calculates the distance between each individual and the booths spread over the area, and Equation (16) calculates the profile similarity between individuals and job fair booth profiles as a fitness function used in each iteration of Equation (17). With all the equations and cycles that will be run based on the fixed parameters, a comparison will be made with the PSO.

2.3. Evaluation

The results of the experimental phase will be evaluated to determine which algorithm is most suitable for the scenario used. The suitability of this model is used to see how natural the movement of each NPC that spawns and walks toward each destination is. In the course of the NPC, how long it takes to explore (roam the job fair exhibition) and exploit the destination (stay in the booth) will be measured. This experiment was compared with 30 iterations of 100 populations and the fitness function in Equation (17).

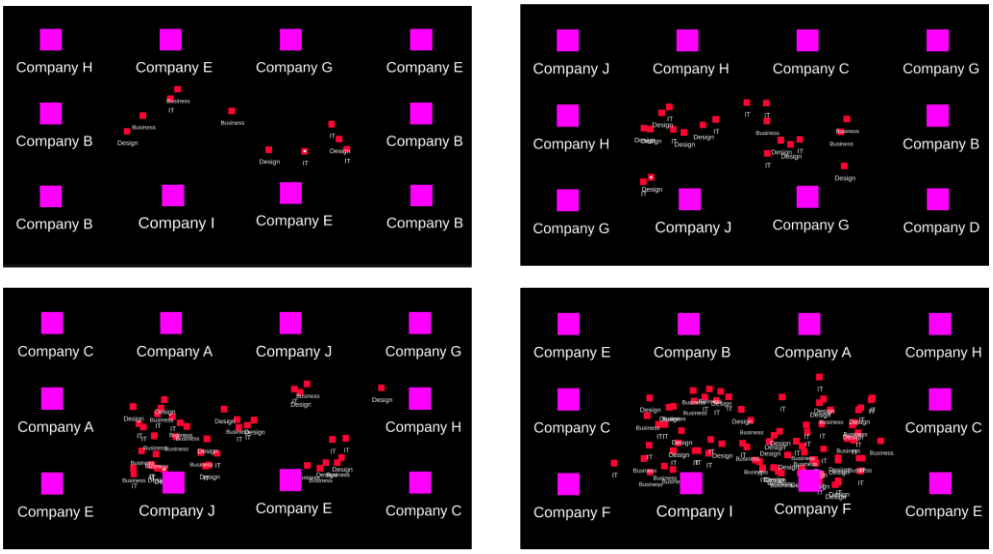

Fig. 3. Scenario Using Population

Fig.3 shows a scenario with several populations: the first with 10 populations, the second with 20 populations, the third with 40 populations, and the last 80 populations. With some of the pictures above, it can be seen that the movements of each individual will first polarize to find food and stay away from enemies, then will continue to approach the food closer and closer and reach the food, represented by the fair job booth. For the movement itself, sometimes it still looks a little stiff because the random value given will determine the direction of the vector.

Table.3 The experimental results use the DA algorithm

Population	*Trial*	*Time for all agents to reach the target (s)*	*Average time (s)*
10	1	8	
10	2	20	18.3
10	3	17	
20	1	30	
20	2	25	33
20	3	44	
40	1	46	
40	2	49	48.3
40	3	50	
80	1	55	
80	2	52	56
80	3	61	

Table 3 shows the results of calculating how long each individual can reach the goal in the unity program with a waiting time of 1 second. The trials carried out several times gave entirely satisfactory results because it was faster than PSO, but it could only reach one goal and still needed development in further research into multi-objectives so that not only one booth was approached by all fair job attendees.

3. Results and Discussion

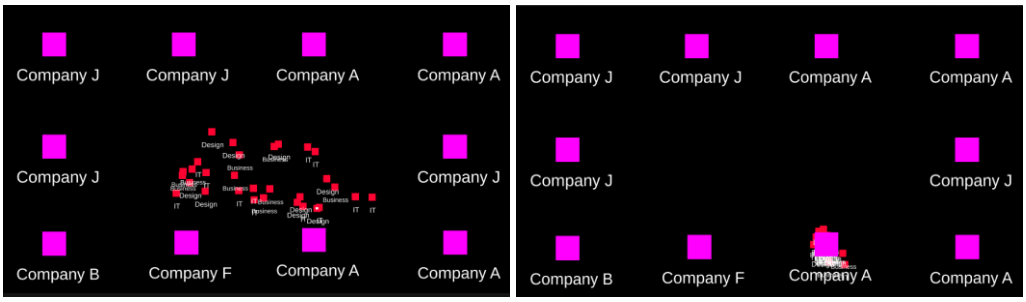

Fig. 4. Konvergen Result

Fig.4 shows when all individuals will move toward the food, namely the fair job booth, by calculating the shortest distance and profile that matches the individual represented as a fair job visitor.

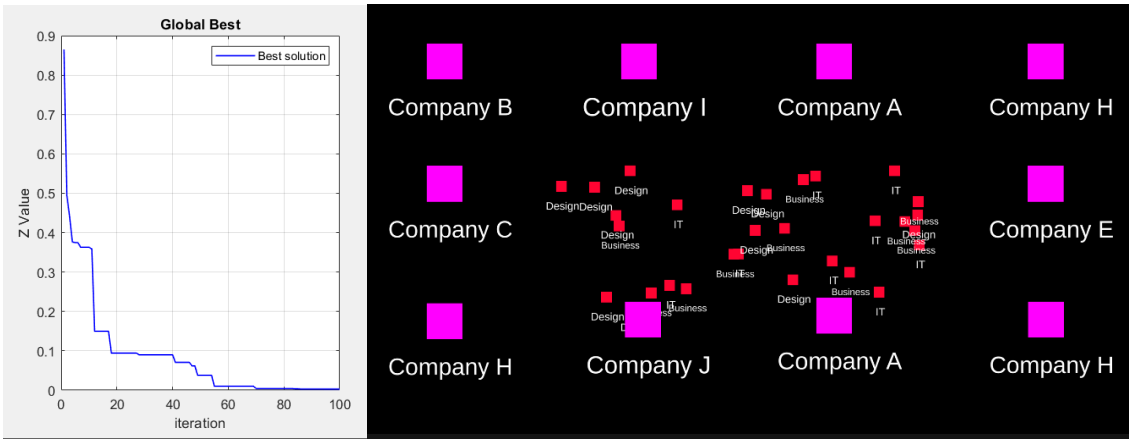

Fig. 5. The results of the PSO algorithm use Matlab and Unity

Fig.5 on the left is the result of using the PSO algorithm in the Matlab program, and the results show convergence at iteration 70, and on the right is a visualization in the unity program in 2 dimensions or top view.

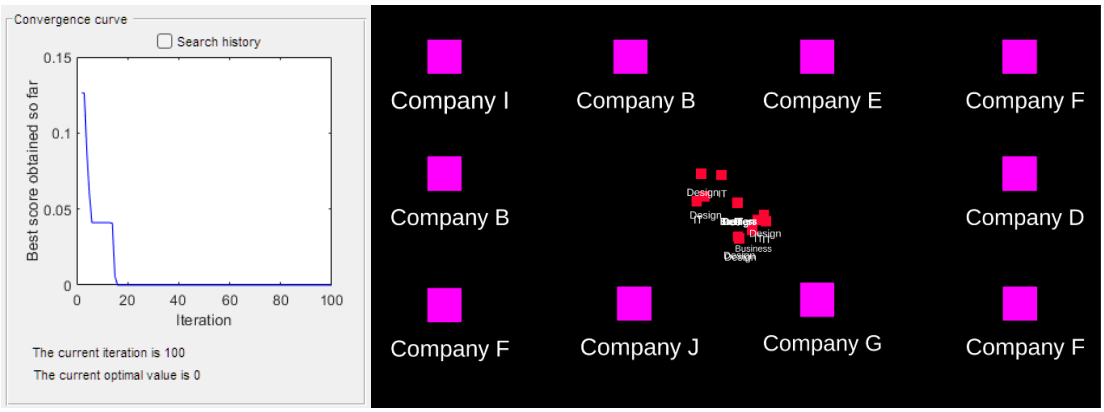

Fig. 6. The results of the DA algorithm use Matlab and Unity

Fig. 6 on the left shows the results of processing the fitness function using the Matlab program, and on the right, the visualization results in Unity in the two dimensions seen above. The results show that the success of DA in solving problems faster than PSO in Fig.3 can be seen from the number of iterations needed to converge. The DA algorithm only requires less than 20 iterations to converge compared to the PSO, which requires 70 iterations.

4. Conclusion

After an experiment comparing PSO with DA, it was found that DA is a more optimal algorithm than PSO and provides a stronger immersive side due to the calculation of neighbors and random movements from Levy's algorithm. However, this experiment still uses a single objective to achieve the goal, and for further research, it will be made with multi-objectives to determine more than one goal and not crowd into one place. Besides that, it will be implemented with 3-dimensional assets in a virtual environment.

References

[1] M. E. Latoschik, F. Kern, J. P. Stauffert, A. Bartl, M. Botsch, and J. L. Lugrin, "Not Alone Here?! Scalability and User Experience of Embodied Ambient Crowds in Distributed Social Virtual Reality," *IEEE Trans. Vis. Comput. Graph.*, vol. 25, no. 5, pp. 2134–2144, May 2019, doi: 10.1109/TVCG.2019.2899250.

[2] C. Kiourt, G. Pavlidis, A. Koutsoudis, and D. Kalles, "Multi-agents based virtual environments for cultural heritage," *ICAT 2017 - 26th Int. Conf. Information, Commun. Autom. Technol. Proc.*, vol. 2017-December, pp. 1–6, Dec. 2017, doi: 10.1109/ICAT.2017.8171602.

[3] A. A. Owaidah, D. Olaru, M. Bennamoun, F. Sohel, and R. N. Khan, "Modelling Mass Crowd Using Discrete Event Simulation: A Case Study of Integrated Tawaf and Sayee Rituals during Hajj," *IEEE Access*, vol. 9, pp. 79424–79448, 2021, doi: 10.1109/ACCESS.2021.3083265.

[4] S. Mirjalili, "Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems," *Neural Comput. Appl.*, vol. 27, no. 4, pp. 1053–1073, May 2016, doi: 10.1007/S00521-015-1920-1.

[5] J. Kennedy and R. Eberhart, "Particle swarm optimization," *Proc. ICNN'95 - Int. Conf. Neural Networks*, vol. 4, pp. 1942–1948, doi: 10.1109/ICNN.1995.488968.

[6] W. Hurst and R. Geraerts, "Augmented and Virtual Reality Interfaces for Crowd Simulation Software-A Position Statement for Research on Use-Case-Dependent Interaction," *2019 IEEE Virtual Humans Crowds Immersive Environ. VHCIE 2019*, May 2019, doi: 10.1109/VHCIE.2019.8714733.

[7] C. W. Reynolds, "Flocks, Herds and Schools: A Distributed Behavioral Model," SIGGRAPH Comput. Graph., vol. 21, no. 4, pp. 25-34, Aug. 1987, doi: 10.1145/37402.37406.

[8] X. S. Yang and S. Deb, "Cuckoo search via Lévy flights," *2009 World Congr. Nat. Biol. Inspired Comput. NABIC 2009 - Proc.*, pp. 210–214, 2009, doi: 10.1109/NABIC.2009.5393690.