Budapest University of Technology and Economics
Faculty of Electrical Engineering and Informatics

Aalborg University
Department of Electronic Systems

COOPERATIVE BEHAVIOUR AND COMMUNICATION

Ph.D. Thesis Booklet

László Blázovics

Advisors:
Dr. Hassan Charaf, Ph.D.
Dr. Frank H. P. Fitzek Ph.D.

BUDAPEST UNIVERSITY OF TECHNOLOGY AND ECONOMICS
DEPARTMENT OF AUTOMATION AND APPLIED INFORMATICS
&
AALBORG UNIVERSITY
DEPARTMENT OF ELECTRONIC SYSTEMS

Budapest, 2014
László Blázovics

Budapest University of Technology and Economics
Faculty of Electrical Engineering and Informatics
Department of Automation and Applied Informatics

1117 Budapest, Magyar Tudósok körútja 2.

E-mail: laszlo.blazovics@aut.bme.hu

Advisors:
Dr. Hassan Charaf, Ph.D.
Dr. Frank H. P. Fitzek Ph.D.
Cooperative behaviour and communication

1 Introduction and Motivation

The distributed coordination and control of a group of decentralised autonomous artificial entities (i.e. robots) is a widely studied problem by various fields like engineering, artificial intelligence, robotics and mathematics. These models were inspired mostly by the biological swarms, like ant colonies, bird flocks, and other herds of animals. Although most researches were focusing on the understanding of such systems, there are already efforts which dealing with the adoption of such biological phenomenon in an artificial environment. This was allowed because not only the computational capacity but also the sensor components and communication modules became affordable for such large scale systems which consist of many entities.

However in order to adapt such behaviour in a real system, the proposed solutions should be validated through theoretical studies and simulation scenarios. Besides the validation of a given solution, these mathematical models are also useful to give theoretical boundaries for certain well studied problems with different conditions. The most examined problems are the gathering and the generation of different formations. These problems were solved by many continuous and discrete time models, in which the entities were modelled as points or small circle shaped objects. The most common way of controlling such artificial systems in these theoretical models is the usage of artificial potential fields. These virtual force based approaches handle the environment as a collection of objects where each unit has a unique potential field which attracts or repulses the other objects.

Through the potential field based continuous models are useful in the theoretical validation of the artificial swarms, the implementation of such solutions in real systems is often not manageable. The main drawback of such models is the fact that each entity of the group affects all their groupmates and the generated forces depend on the position of the other entities. This means that each entity should continuously know the exact position of the others in the whole group, which is hard to implement in a real life systems.

Therefore there are other types of theoretical models, which divide the life cycle of the entities into discrete time phases. The most common model is the look-compute-move model, in which the entity first monitor the area (look), then based on the gathered information calculates its next movement (compute) and finally proceeds to the desired destination (move). This discrete model has three variants depending on the synchronisation level of the entities life-cycle: There are fully-synchronous, semi-synchronous and fully-asynchronous models. Due to the discrete life-cycle, these models could be implemented into a real life system easily.

As the theoretical results became adaptable in real-life scenarios the need for robust solutions became significant. In order to generate fault tolerant behaviours, many research were using oblivious entities. By the use of these memory-less entities the addition, the removal and the replacing of such entities of the swarm became transparent to the whole system. Besides the obliviousness, the use of limited sensing capability – instead of the infinite sensing range – became popular too. The use of sensors with limited sensing range made the adaptation of the algorithms easier. However these limitations made the theoretical validation of the solutions harder.

Besides the swarm robotics, the adaptation of such models became popular also in the field of sensor self-deployment. The proper deployment of large scale sensor networks was always a complicated task. Furthermore there are situations – like a nuclear catastrophe or any kind of natural disasters –, where the deployment cannot be managed by humans. Therefore the automation of the sensor deployment became a popular research area and
thanks to that these autonomous sensors do not differ significantly from the entities of the artificial swarms, the adaptation of their cooperative behaviour can be made easily: mobile sensors have "hearing", while mobile robots have "vision" [LNSRS10].

Due to the limited capabilities of the swarm entities, there are situations which are hardly solvable by these units. The most common problem with the usual solutions is the lack of information sharing. It should be noted that the addition of communication capabilities for the swarm was already made by many researches and regarding the sensor systems, this capability is essential in that use-case. However these communication layers were basically used for centralised controlling which is decreased the robustness of these systems. In contrast with the centralised concept, the addition of peer-to-peer communication capabilities for such systems does not affect their robustness because it does not harm the individuality of the entities. Nevertheless it increases the performance of the overall swarm because the entities will be able to share their gathered information among each other.

The goal of this research is to design and evaluate novel solutions by which a group of individual entities is able form a fully decentralised collective in order to behave like an artificial swarm. The following open questions are addressed by the thesis:

- How to develop cooperative distributed behaviours like area discovery and surrounding for artificial swarms?
- How can a group of individual entities monitor a given area in order to prevent additional intruders to pass through it?
- How to solve the intruder problem with individual, oblivious entities by the use of a surrounding algorithm?
- What kind of additional rules and prerequisites can make the Greedy Rotation Greedy (GRG) based sensor self-deployment faster while keeping the same Focused Coverage (F-Coverage) ratio?
- How can a group of individual entities share group related data among each other in an effective way in order to enhance their performance?

2 Methodological Summary

The outlined open issues have defined the initial direction of my research. The first steps for both the cooperative robotics and the sensor self-deployment related topics were to analyse the already working solutions and to find potential extension possibilities. Therefore I have used multiple environment such MatLAB and V-Rep. However, the most part of the simulations and analysis were made in my custom environment, called Discrete Swarm Simulator (DSS). This custom environment was used also during the evaluation of my novel results.

These experiments and measurements revealed what the key issues with target (or POI) surrounding and cognitive capabilities are and what directions further research should take. The next step was designing the new behaviours, and algorithms. My thesis introduces the Multi Orbit Surrounding and the mGRG algorithms for solving the fast surrounding problem and presents the cognitive swarm concept. The evaluation of these solutions forms a principal part of the research. First of all, the impact of the
new concept and algorithms was investigated. Besides the theoretical validation of the proposed concept and algorithms simulations were also used for the same purpose. During my research, the suitability for real-world applications was an important objective. Therefore some parts of my research were already implemented and evaluated in commercial applications.

3 Novel Scientific Results

The scientific results of my research are summarized in three theses. Thesis I presents an area discover concept and proposes a technique for reducing energy consumption for individual swarm entities. Besides, Thesis I also discusses the Multi Orbit Surrounding (MOS) algorithm, by which a group of individual entities are able to encircle a given target. Thesis II presents a modified version of the Collision Avoidance version of the Greedy Rotation Greedy (GRG/CV) algorithm of Li et al. for solving the Focused Coverage (F-Coverage) problem. Thesis III introduces the area surveillance problem, and by the use of the problem it presents the cognitive swarm concept, by which the members of the swarm will be able to consciously share their knowledge among their swarm-mates.

THESIS I

Cooperative Robotics - Discovery and Surrounding Algorithm

I have designed a hybrid concept for distributed area discovery for a group of semi-individual entities. I have proposed a technique for reducing energy consumption where I have shown that if the robots avoid performing the "stop-start" phases of the translation process, they consume less energy. I have presented the fully-synchronous, discrete time, continuous space Multi Orbit Surrounding (MOS) algorithm, by which a group of individual entities are able to encircle a given target. I have proven that this solution is able to encircle a slowly moving target within a predefined time constant by a group of decentralised, oblivious entities who are unable to communicate with each other and rely only on their limited sensors and the position of the target. The presented theoretical results were also evaluated through simulations.

Related publications: [3] [10] [11] [12] [14] [15] [16] [17] [18]

Thesis I is covered by Chapter 4 in the dissertation.

Subthesis 1.1.: Area Discovery

I have introduced a distributed area discovery algorithm for a group of individual entities who have collective global knowledge. I have presented a method for reducing the energy consumption during the discovery process.

The main goal of the proposed concept is to create a map from a given area, which will be used by the members during the exploration. As the entities move along the explored area, they are simultaneously extending this common map until they are unable
Figure 1: Example for area discovery with three robots, where (a) and (b) show the initial and the end setup.

to find unexplored areas. There are already swarm-based discovery solutions, like the idea of [BSK11]. Unlike the definition of dedicated entity types, I rather split the task of the entities into two parts: translation and mapping. This keeps the swarm homogenous while the robots are able to perform multiple types of tasks. During the exploration, each member of the swarm successively repeats the following two processes until they reach the common goal: the full discovery of a given area.

**Translation:** This process is responsible for moving the robots from one point to another. Before the effective translation, each robot - based on the internal map - determines the next target (or destination) point which should be reached after the translation. Each target has a dedicated direction where the target points. After the robot reached the target, it should rotate towards that direction.

**Mapping:** After the robot reached the destination point, it starts the area discovery. In this process, the robot uses its built-in sensors (i.e. mounted cameras) for gathering information from that area, where the current place points. With the gathered information, the robot extends the common map and switches back to the translation phase.

The key part of the overall discovery process is the determination of the next targets. In the next section I will introduce an algorithm that gives a baseline solution.

It should be declared that this algorithm does not give the optimal solution, especially if all targets are known before the first iteration - as it was declared above. This is because the problem is analogue to the Multiple Traveling Salesmen Problem (mTSP) which is NP-complete [KA10]. Nevertheless, as the forecast level - the amount and the distance of available targets - decreases, the result of the algorithm may converge to the optimal solution.

**Proposition 3.1.** Those discovery solutions where the entities are able to move along and map the discoverable area continuously require less energy than those ones in which the robots should repeatedly start and stop for making their measurements.

**Subthesis 1.2. Multi Orbit Surrounding**

I have presented the discrete time, fully-synchronous Multi Orbit Surrounding (MOS) algorithm by which a group of individual, oblivious entities with limited sensing capabilities are able to surround a moving target. I have proven that the surrounding process always finishes in polynomial time and I have also gave an upper bound for the surrounding time.
The Surrounding Concept

If multiple entities are trying to capture a moving target one of the most relevant tactics is the surrounding. Most animal groups who are hunting in a group - like wolf packs - are following this strategy. Therefore, the main idea of the presented mechanism is to define a trajectory around the target on which the entities should take place.

However, as in real-life situations, the target is trying to avoid from the interception. In order to assure that the target cannot escape from the circle of the entities - i.e. the circle is not uniformly filled and there are holes in it - , they should move around it. In order to minimise the possibility of the inter-agent collision, a heading direction was defined for the trajectory. Nevertheless, by the introduction of this strict direction rule on the desired trajectory prohibits the use of a usual pincer movement.

In order to solve this issue, I have introduced the multi orbit surrounding mechanism. I have defined multiple orbital trajectories around the target which are moving with it. A simple example can be seen in Figure 2.

Each trajectory has a heading direction. This serves two purposes:

- First this mechanism accelerates the surrounding process.
- Second it reduces the probability of inter-agent collision, because all agents are moving in the same direction.

Each neighbouring trajectory pair has different heading direction, which implies that there should be some distance between them in order to minimise the collision probability. The basic behaviour of the entities between two trajectories is a radial movement towards the target. There are two types of trajectories.

The first type is the primary trajectory ($T_{primary}$). This is the nearest trajectory around the target. The main goal of the entities is to put themselves into orbit on this trajectory. Whenever an entity has reached this trajectory, it should stop all radial movement and should start to move around the target in the given heading direction.

The second type of trajectories is the secondary trajectory ($T_{secondary}$). These trajectories are more distant than the primary. If an entity is passing through one of these
trajectories during the surrounding process and it is sensing another entity in front of it, it should put itself onto orbit on the current trajectory. By doing this not only the collision will be avoided but the surrounding process will be accelerated too.

However if the entities are not staying on any of the above introduced trajectories, it is not allowed to make tangential movements around the target.

In order to avoid inter-agent collision, the entities also have potential fields around themselves with significantly stronger repulsion component than the attraction of the target.

By defining this trajectory system, those entities who are coming from the same direction towards the target are enabled to make a classical pincer movement and surround the target. The generated behaviour is the following:

If the current position of the robot is on $T_1$ or a neighbour with higher priority prohibits to move towards the target, circulate in the corresponding direction around the target on $T_i \mid i > 2$; in all other cases move directly towards the target.

The pseudocode of the algorithm can be seen on Algorithm 1.

**Algorithm 1 The Multi Orbit Surrounding Algorithm**

```plaintext
Input: $H$ all members of the swarm, $T$ target, $Tr_i \mid i \in \mathbb{Z}$ trajectories of $T$, $R_i \mid i \in \mathbb{Z}$ members of $H$
loop
    for $i = 1$ to $|H|$ do
        face to the target
        if $R_i$ is on $Tr_0$ then
            rotate on the current orbit with the heading direction of $Tr_0$
        else if $R_i$ is on $Tr_i$ then
            if There is no other robot in front then
                move towards the target
            else
                rotate on the current orbit with the heading direction of $Tr_i$
            end if
        else
            move towards the target
        end if
    end for
end loop
```

During the theoretical validation the following **Assumptions** were used:

1. Given $n$ robots, modelled as points in the Euclidean plane $\mathbb{R}^2$.
2. The robots use a common coordinate system.
3. All robots know the position of the target $t$ and the target is placed at the origin.
4. The movements of the robots are divided into discrete steps.
5. In each step a robot can move a unit distance in the direction determined by the potential field or stay in place if this movement is prohibited by another robot.
6. The Euclidean distance \( d(u,v) \) between each pair of robots \( u,v \) must be at least some constant \( d_{\text{min}} \leq 1 \).

7. The radius \( r_1 \) of the primary orbit \( T_1 \) is 1 and the distance between neighbouring orbits is 1, i.e. \( r_{i+1} - r_i = 1 \), \( i = 1, 2, ... \), where \( r_i \) is the radius of the \( i^{\text{th}} \) orbit \( T_i \).

8. The sensing range of the robots is at least \( d_{\text{min}} + 2 \). Thus, each robot which could get closer than \( d_{\text{min}} \) after one time step is within the sensing range.

9. The robots are able to switch between neighbouring orbits within one time step.

**Proposition 3.2.** After \((N + 1)/2 \times D \) time step the primary trajectory \( T_0 \) will always be filled and therefore the target will be encircled, where \( N \) is the maximum number of entities who are able to take place on \( T_0 \) and \( D \) is the sum of distances of the robots who will reside on \( T_0 \) to the target in the starting configuration.

## Thesis II

**Sensor Self-Deployment - Extending the Focused Coverage**

I have presented the modified Greedy Rotation Greedy (mGRG) algorithm to solve the Focused Coverage (F-Coverage) problem in self-deploying mobile sensor networks. The algorithm is a modified version of the GRG/CV algorithm by Li et al. [LFSS11]. I have proven that the presented algorithm always guarantees that the sensor nodes enclose the POI without sensing holes in \( O(D) \) time step, where \( D \) is the sum of distances of the nodes from the POI in the initial configuration. This significantly improves the previous bound on the coverage time. The theoretical results are also validated by simulations. Finally both the theoretical results and the simulations showed that the mGRG algorithm results in a faster coverage than the GRG/CV.

Related publications: [5] [13] [19]

Thesis II is covered by Chapter 5 in the dissertation.

**Subthesis 2.1.: The mGRG concept**

Based on the Greedy Rotation Greedy (GRG) algorithm, I have designed the modified GRG (mGRG) algorithm by which a group of individual, oblivious sensors can cover a given area around a given centre point without sensing holes and by maximised Focused Coverage (F-Coverage). I have proven that that the mGRG algorithm guarantees that after \( O(D) \) steps each node reaches its final layer, where \( D \) is the sum of initial hop distances of the nodes from the POI (Point Of Interest) in the TT (Equilateral Triangle Tessellated Graph). I have validated the theoretical results through simulations.

The main ideas of the presented concept are the following. First, each hexagonal layer is assigned a heading direction, in such a a way that any two neighbouring layers have opposite heading direction. For example, odd layers have counterclockwise and even layers have clockwise heading direction. This solution is similar to the multi orbit solution which was presented in Section ??.
step, it moves around the POI in the given heading direction. Second, if a greedy advance movement and a rotation movement target the same vertex, the rotational movement gets higher priority. This principle will ensure that each sensor can keep moving in each time step, since rotation will be always possible. The sensors move straight towards the POI until they reach the innermost hexagonal layer. This is the primary trajectory ($T_1$).

Similarly to the base GRG, if a node is unable to get closer to the POI – because of another node is in front of it or it has reached the innermost layer – it should rotate on the current layer. If the node is able to move to an inner layer, it should check whether an other node is trying to get to the same place. An example can be seen on Figure 3.

**Priority rules**

Consider a vertex $x$ on the layer $T_i$ (The vertex $x$ can be occupied or not at the moment. If it is occupied, the sensor occupying it will move in the next step.) The vertex $x$ has at most four neighbouring vertices from which it can be occupied in the next step if $x$ is a corner vertex, and at most three, otherwise. One such neighbouring vertex is on the same layer $T_i$ and at most three vertices on the next higher layer $T_{i+1}$. Regarding $x$ the highest priority is assigned to the neighbour vertex on $T_i$. The heading direction $T_{i+1}$ defines an order on the neighbour vertices on $T_{i+1}$. The first one in this order gets the second highest priority, the second the next highest, etc. For example, in Figure 3 vertex $p_1$ can be occupied from vertices $q_1$, $q_2$, $q_3$, $p_2$. The priority order from highest to lowest is $p_2$, $q_3$, $q_2$, $q_1$. A sensor $u$ obtains the same priority than the vertex currently occupied by $u$. A sensor $u$ can occupy a vertex $x$ if no other sensor resides on a vertex with higher priority regarding $x$. Note that each sensor is aware of the sensors that can occupy the same vertex, since they are in the 2-hop neighbourhood of each other. Thus, each sensor can decide locally whether it has the highest priority among them.

If a sensor $u$ is equally far from two vertices closer to the POI than $u$ (like $q_1$ from $p_0$ and $p_1$ in Figure 3) and the heading direction of $u$ is counterclockwise (clockwise), then
Cooperative behaviour and communication

the vertex left (right) from the direction of the POI is preferred. If there is another sensor with higher priority regarding this vertex, then \( u \) chooses the other. If this vertex can also be occupied by a higher priority sensor, then \( u \) must rotate.

These rules imply that each sensor is either moving towards the target or rotating around it, they never stay on the same place in the next time step. They also imply that a node must know the 2-hop neighbourhood in order to avoid collision and detect an occupiable vertex in the next inner layer.

**Proposition 3.3.** After \( O(D) \) time steps all inner layers are filled, where \( D \) is the sum of initial hop-distances of the sensors to the POI.

**Subthesis 2.2.: Decreased coverage time**

I have proven that the mGRG algorithm performs the coverage process faster than the Collision avoidance version of the original GRG (GRG/CV). I have validated the results via simulations.

The simulation results, which demonstrate the enhanced surrounding speed of the algorithm, can be seen on Figure 4.

**Proposition 3.4.** The mGRG performs the coverage process faster than the original GRG/CV algorithm.

**Thesis III**

**Cooperative Communication - Cognitive Swarms**

I have introduced the concept of cognitive swarm by which a group of individual robots will be able to become a real collective entity which is able to make inter cognitive communication channels with other high level cognitive entities like humans. In order to evaluate the benefits of the cognitive swarm concept I have presented the area surveillance problem in which a swarm of mobile robots had to guard a given area by intercepting eventual intruders. I have formalised the area surveillance problem and I have presented two basic behaviour set-based solutions for that. I have given a discrete interpretation of the generated Area Surveillance solutions. I have evaluated the effectiveness of both behaviour based discrete solutions through theoretical analysis and through multiple simulation scenarios. I have also extended these solutions with cognitive infocommuniation aided capabilities and I have showed through theoretical and simulation results that due to these capabilities it is more effective than a usual swarm.

Related publications: [4] [20] [21] [13]

Thesis III is covered by Chapter 6 in the dissertation.

**Subthesis 3.1.: The Area Surveillance**

I have introduced the Area surveillance problem and I have proposed two basic behaviour set-based solutions for that. I have proven that by the use of the Discrete Following Rule the entities of a swarm is able to surround a moving target by the use of the discrete variant
Figure 4: Coverage times for mGRG GRG/CV, (a) fixed size network, variable size dropping area, (b) variable size network fixed size dropping area

of the MOS algorithm. I have shown through simulation results, that both solutions are able to intercept a given target on a bounded area.

In order to design the fundaments of such distributed algorithms I have used the basic behaviours of Mataric [Mat95] like building blocks. The proposed surveillance concept is divided into two parts.

**Dispersion:** The first part or phase of the area surveillance is the dispersion of the entities from a so-called base. It is evident that the dispersion will emerge the possibility of the detection of an intruder by an entity. However it is also important to avoid creating sensing holes between the entities at the end of the dispersion. As it was depicted by Gage [Gag92] the maximisation of surveilled area and the minimisation of sensing holes cannot be achieved simultaneously. It is only available in optimal situations e.g. when the size of the patrolling area is small. Therefore, it should be decided whether to maximise the number of the detected enemies over a wide area or to minimise the probability of leaving any intruder undetected. In this concept, the minimisation of sensing holes was chosen.

**Interception:** The second part is the interception of the intruder or the target. In contrast with the dispersion, the proposed surrounding method inquires much more cooperation than the dispersion; therefore, two different solutions, a baseline and a multi orbit surrounding based, will be presented.
Algorithm 2 Dispersion

loop
    if one or more entities are within $d_{\text{disperse}}$ then
        move away towards the least dense area
    end if{this is uniform for all of the entities}
end loop

If an entity detects the target, it automatically switches from dispersion to interception state and in case of losing the target it will switch back to dispersion mode.

In the basic behaviour set inspired Baseline algorithm, we distinguish between two behaviours. The first one is a gathering- or homing-like motion, in which the entity is trying to move directly towards the target. The entity follows this behaviour until there is no other entity in front of it or until the distance between the target $T$ and itself is greater than a predefined $d$ distance. If this homing motion is not possible, the entity should circulate around the target in a predefined direction until the homing becomes possible again. This is the circulating behaviour. In order to avoid collision situations, circulation is only allowed on dedicated orbits around the target. The radius of the first orbit should be $d$. After translating this into the language of basic behaviours we get the desired behaviour of target interception which can be seen in Algorithm 3.

Algorithm 3 Baseline target interception

loop
    if current position is on the first orbit or a neighbour prohibits to move towards the target then
        Circulating
    else
        Homing
    end if
end loop

Although the Multi Orbit Surrounding based algorithm performs better than the baseline algorithm, there is only one significant difference between them. This is the predefined heading direction of the trajectories. The basic behaviour based interpretation can be found in Algorithm 4.

Algorithm 4 Multi-orbit target interception

loop
    if current position is on the first orbit or a neighbour prohibits to move towards the target then
        Circulating in the corresponding direction
    else
        Homing
    end if
end loop

Both for easy evaluation and real life testing I have used a discrete representation of the above presented behaviours. This discrete representation used a fully synchronous
model where both the target and the entities are able to move only on the edges of a TT graph. The following rules were made in order to adapt the generic algorithm into the discrete environment.

1. In each time step all entities are able to move from their residing vertex to one of its neighbouring vertex.

2. The minimum distance between two entities during the interception \(d_{\text{intercept}}\) is 1 unit.

3. The velocity of the target \(v_t = v_e/4\). (This means that the target are able to move from its residing vertex to one of its neighbouring vertex in every fourth time step.) Therefore the period time of the entities \(p_e = 1\) while the period time of the target \(p_t = p_e \cdot 4 = 4\). However the algorithm also works with slower target.

4. The distance between neighbouring trajectories is constant 1 unit.

5. The sensing and communication radiuses of the entities are \(r_s = 4\), \(r_c = 8\).

The entities use the \textit{look-compute-move} model. If an entity senses the target, it switches to the interception state and tries to surround it. Nevertheless if an entity is not able to sense the target in the actual time step, it switches back to the dispersion behaviour for that step. In order to adapt the interception algorithms to the moving target in the discrete space environment, the following rule was added to the original behaviours:

\textbf{Discrete Following Rule:} If the entities, who are in interception state, detect that the target moved - take one step - all of them take one step towards the same direction. Although this additional step prevents the entities to perform the main surrounding behaviour for one time step, during rest of the time when the target do not move they are able to resume the surrounding process from where it was left off right before the target moved.

The simulation results of target interception for both the usual and cognitive solutions can be seen on Figure 5.

\textbf{Proposition 3.5.} By using the MOS algorithm and the Discrete Following Rule a group of swarm entities are able to surround a moving intruder in \(\lfloor (6/2 + 1) \cdot D \cdot p_t / (p_t - p_e) \rfloor\) time steps, where \(D\) is the sum of distances of the entities who will reside on the primary trajectory and \(p_t\) and \(p_e\) are the period times of the target and the entities, \(p_e = 1\) and \(p_t = i \cdot p_e\ \text{ where } i \geq 2, i \in \mathbb{Z}\).

\textbf{Subthesis 3.2.: The cognitive swarm}

I have introduced the cognitive swarm concept where the neighbouring entities have peer-to-peer communication channels between each other. I have proven that the communication aided swarms adapt to the changes of their environment faster. I have also proven that in a discrete, ideal, sensing hole free coverage, the entities of a cognitive swarm always surround a intruder in polynomial time. I have given an upper bound for this surrounding time. I have shown via simulation results that the communication aided solution performs better than the communication-less swarms.

In order to acquire and use the knowledge of the other members of the swarm, communication capabilities should be added to the system. When this communication layer
has been established, the swarm will emerge from a group of individual members who are able to sense each other into a higher level entity. This entity will be able to communicate through intra-cognitive channels with other such cognitive entities and it will also be able to communicate with its members via inter-cognitive communication.

In this surveillance example, by the addition of such capabilities the robots will be able to share and propagate the position of the target and they will be able to determine collectively whether the target is successfully intercepted or not. In an usual case, it should be determined by an external observer.

This information-sharing potential is going beyond the benefits of the area surveillance problem, however, it is a good example to demonstrate the benefits.

**Proposition 3.6.** The communication aided, cognitive swarms adapt to the change of their environment faster than those usual swarms that rely only on local sensing.

In the case of the area surveillance problem, this means that the cognitive swarm requires fewer time steps for surrounding the target because it detects the intruder faster than the usual swarm. (See Figure 5.)

**Proposition 3.7.** A group of cognitive swarm entities who are able to cover a given area without sensing holes with a sensing range of $r_s$ are able to surround an intruder by using the Area surveillance behaviour with the MOS algorithm within $\lceil(6/2 + 1) \cdot 10 \cdot r_s \cdot p_t/(p_t - p_e) \rceil$ time steps.

4 Practical Application of the Scientific Results

Related publications: [1] [2] [6] [7] [8]

In order to use the previously presented theoretical results which I have made in the field of cooperation and information sharing I have participated in two application development projects in which I was able to evaluate the fundamental parts of the introduced concepts.

Regarding the network aided cognitive information sharing I have participated in the development of a Network Coding based photo sharing application, the Picture Viewer. The goal of the project was to demonstrate the feasibility and the effectiveness of such
complex communication layer in an embedded environment. Besides the presentation of a working solution, it was also shown that in case of small data size the energy cost of packet coding is less relevant.

For evaluating the effectiveness of cooperation, I took part in the development of a social mobile game, the Gedda Headz in order to validate the benefits of cooperation in a real social community.

5 Related Publications

Book Chapter


Journals


Conference Proceedings


Cooperative behaviour and communication


6 Other Publications


References


