CZECH TECHNICAL UNIVERSITY IN PRAGUE

Faculty of Electrical Engineering

Bachelor Thesis

Petr Všetečka

Navigation and Stabilization of Swarms of Micro Aerial Vehicles in Complex Environment

Department of Cybernetics

Thesis supervisor: **Ing. Martin Saska, Dr. rer. nat.**

Czech Technical University in Prague
Faculty of Electrical Engineering

Department of Cybernetics

BACHELOR PROJECT ASSIGNMENT

Student: Petr Všetečka

Study programme: Open Informatics

Specialisation: Computer and Information Science

Title of Bachelor Project: Navigation and Stabilization of Swarms of Micro Aerial Vehicles in Complex Environment

Guidelines:

1. Student will study the algorithm for stabilization of swarms of unmanned helicopters based on BOIDS model, which is inspired by flocking behaviour in nature [1,2].
2. This method will be extended with possibility to follow a dynamic target.
3. Student will study an algorithm for motion planning in complex environments with obstacles. The algorithm will be integrated into the method for stabilization and control of the swarm.
4. Constraints given by visual relative localization of swarm members [3] and constraints given by using real helicopters will be integrated into the swarm stabilization rules.
5. Student will design and implement a proper heuristic for estimation of time of flight through corridors of different size and a function for more exact estimation of the flight time using motion simulations.
6. The complex system will be verified in numerical experiments in environments with different complexity and its behaviour will be statistically analyzed.

Bibliography/Sources:

[1] Saska, M. - Vakula, J. - Přeučil, L.: Swarms of micro aerial vehicles stabilized under a visual relative localization, Proc. of IEEE International Conference on Robotics and Automation (ICRA), 2014.

[2] Reynolds, C. W.: Flocks, herds, and schools: A distributed behavioral model, in Computer Graphics, 1987, pp. 25–34.

[3] Faigl, J. - Krajník, T. - Chudoba, J. - Přeučil, L. - Saska, M.: Low-Cost Embedded System for Relative Localization in Robotic Swarms. In ICRA2013: Proceedings of IEEE International Conference on Robotics and Automation, 2013.

[4] Krajník, T - Nitsche, M - Faigl, J. - Vaněk, P. - Saska, M. - Duckett, T. - Přeučil, L. - Mejail, M.: A practical multirobot localization system. Journal of Intelligent & Robotic Systems, Volume 76, Issue 3-4, pp 539-562, 2014.

[5] Lee, T. - Leoky, M. - McClamroch, N.: Geometric tracking control of a quadrotor uav on se(3), in 49th IEEE Conference on Decision and Control (CDC), 2010.

Bachelor Project Supervisor: Ing. Martin Saska, Dr. rer. nat.

Valid until: the end of the summer semester of academic year 2015/2016

L.S.

doc. Dr. Ing. Jan Kybic
Head of Department

prof. Ing. Pavel Ripka, CSc.
Dean

Prague, February 10, 2015

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra kybernetiky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: Petr Všetečka

Studijní program: Otevřená informatika (bakalářský)

Obor: Informatika a počítačové vědy

Název tématu: Navigace a stabilizace roje bezpilotních helikoptér ve složitém prostředí

Pokyny pro vypracování:

1. Student nastuduje algoritmus pro stabilizaci roje bezpilotních helikoptér založený na modelu BOIDS, který je inspirován chováním rojů v přírodě [1,2].
2. Rozšíří tuto metodu o možnost sledování dynamického cíle.
3. Nastuduje algoritmus pro plánování pohybu robotů ve složitých prostředích s překážkami a integruje jej do vyvinuté metody pro stabilizaci a navigaci roje.
4. Do pravidel rojového chování integruje omezení daná vizuální relativní lokalizací členů roje [3] a omezení daná použitím reálných helikoptér.
5. Navrhne a implementuje heuristiku umožňující odhad doby letu roje v závislosti na tvaru prostředí a navrhne a implementuje funkci umožňující tuto dobu odhadnout přesněji pomocí simulace.
6. Výsledný systém verifikuje numerickým experimentem v prostředích různé složitosti a statisticky analyzuje jeho chování.

Seznam odborné literatury:

[1] Saska, M. - Vakula, J. - Přeučil, L.: Swarms of micro aerial vehicles stabilized under a visual relative localization, Proc. of IEEE International Conference on Robotics and Automation (ICRA), 2014.
[2] Reynolds, C. W.: Flocks, herds, and schools: A distributed behavioral model, in Computer Graphics, 1987, pp. 25–34.
[3] Faigl, J. - Krajník, T. - Chudoba, J. - Přeučil, L. - Saska, M.:Low-Cost Embedded System for Relative Localization in Robotic Swarms. In ICRA2013: Proceedings of IEEE International Conference on Robotics and Automation, 2013.
[4] Krajník, T - Nitsche, M - Faigl, J. - Vaněk, P. - Saska, M. - Duckett, T. - Přeučil, L. - Mejail, M.: A practical multirobot localization system. Journal of Intelligent & Robotic Systems, Volume 76, Issue 3-4, pp 539-562, 2014.
[5] Lee, T. - Leoky, M. - McClamroch, N.: Geometric tracking control of a quadrotor uav on se(3), in 49th IEEE Conference on Decision and Control (CDC), 2010.

Vedoucí bakalářské práce: Ing. Martin Saska, Dr. rer. nat.

Platnost zadání: do konce letního semestru 2015/2016

L.S.

doc. Dr. Ing. Jan Kybic
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 10. 2. 2015

Prohlášení autora práce

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne ....

Acknowledgement

I would like to thank my family for their continuous support and I would also like to thank my supervisor Ing. Martin Saska, Dr. rer. nat. for his guidance and advices.

Abstract

This thesis deals with navigation of swarms of MAVs in complex, mainly indoor, environments. It is based on distributed algorithm able to reach a static goal while avoiding spherical obstacles. This algorithm is extended by ability to follow a dynamic goal moving on a predetermined path and allows deployment of the system in complex environments. It uses convex polyhedra to form the obstacles and the GJK algorithm to compute the distance between MAVs and the obstacles. Finally, complex behaviour of the swarm, where individuals are controlled by the developed local rules, is analysed while passing through a narrow alley. It provides means to estimate possible outcome, which is crucial for high level planning.

Keywords: micro aerial vehicles, autonomous navigation, robotic swarms

Abstrakt

Tato práce se zabývá navigací rojů bezpilotních helikoptér ve složitých prostředích, zejména uvnitř budov. Je založena na distribuovaném algoritmu pro dosažení statického cíle a schopném vyhnout se kulovým překážkám. Do původního algoritmu přidává schopnost sledování složitější cesty pomocí pohyblivého cíle a umožňuje nasazení systému v komplexních prostředích. Tato prostředí vytváří z konvexních polygonů a pomocí GJK algoritmu počítá vzdálenosti mezi helikoptérami a překážkami. Na závěr analyzuje chování roje, ve kterém se jedinci chovají podle vyvinutých pravidel, při průletu úzkým průchodem. Předkládá prostředky pro odhad výsledku před vlastním výpočtem, což je nezbytný prvek pro pokročilé plánování.

Klíčová slova: bezpilotní helikoptéry, autonomní navigace, hejna robotů

Contents

List of Figures

List of Tables

1 Introduction

The main objective of this Bachelor thesis is to design, implement and verify an algorithm for navigating a swarm of quad-rotors through a complex environment. This algorithm will be built upon the algorithm enabling an escape behaviour of the swarm described and implemented by Jan Vakula [4].

The algorithm in [4] is capable of navigating a swarm of unmanned micro aerial vehicles (MAVs) through specific environment. This algorithm is based on a model of flocking behaviour of animals described in [2], and is able to keep the swarm together while avoiding obstacles. This algorithm however is limited to following a straight path leading to a static goal and to environment with constant size spherical obstacles.

This thesis tries to build a robust system capable of navigating a swarm of MAVs through a complex environment and studies the swarm behaviour in narrow corridors. First it extends the original approach using a static goal into a new approach enabling to consequently follow a dynamic set of goals as described in chapter 3.1. The requirement for complex environment led to implementation of polyhedral obstacles. This is described in chapter 3.2. With polyhedral obstacles, a new way of collision detection needed to be implemented. Since this was already a topic of interest in the previous works like [5], it was decided to choose GJK algorithm as the collision detection system for this work. Details on this can be seen in chapter 3.3.

Then it was needed to add restrictions given by MAVs to the system. This consists of the fact that quad-rotors cannot overlap each other which is discussed in chapter 4.1, and also the fact that one quad-rotor cannot fly above another, since the one below would lose lift and fall to the ground. This is the topic of chapter 4.2. It was also required to implement restrictions of relative visual localization system into the methodology being developed here which is the aim of chapter 5.

Finally the developed system was verified in numerous simulations. A lot of simulations were run with different sized swarms in various alleys to thoroughly test behaviour of the swarm in similar situations. These simulations were analysed and results were formulated (see chapter 6.1). The restrictions of relative localization were implemented and verified by numerous tests in chapter 6.2. Then the system was tested in different complex environments to verify its variability and this is described in chapter 6.3.

2 Preliminaries

In this chapter, quad-rotor platform and model of its behaviour as part of a swarm studied and implemented in [1, 4] will be described briefly to provide reader with basic understanding. The reader can look into both [1, 4] to gain a deeper insight into this problematic.

2.1 Quad-rotor

Quad-rotor, or quadcopter, is a multirotor helicopter with four rotors placed in a square pattern, example on Figure 1. Two rotors spin clockwise and the other two counter-clockwise. It is type of unmanned micro aerial vehicle (MAV). These MAVs have six degrees of freedom: three positional and three rotational. Control over their flight is managed only by adjusting the speed of each of its four rotors.

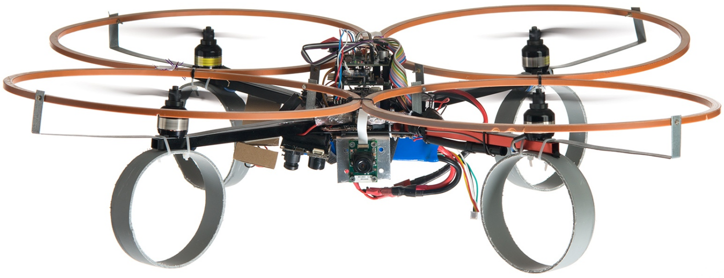

Figure 1: Example of quad-rotor. Source: mrs.felk.cvut.cz

Quad-rotors have several advantages over other aerial vehicles:

- It is capable of vertical take-off and landing (VTOL).
- It is easy to stabilize and has no limitation on minimum flight speed (unlike plane for example), therefore it can be stabilized on a certain position.
- It is capable of precise movement (depending on its controller and/or hardware)
- It has a simple mechanical structure reducing both price and repairs.
- Using four propellers instead of one allows each rotor to have smaller diameter which greatly reduces the amount of kinetic energy stored in the propellers and makes whole machine much safer and more resistant in case of crash. [4]

Because of these advantages and advancement in industry, quad-rotors have become cheap and popular to use for research and for fun as well. There are also a lot of possibilities for research, for instance cooperative reconnaissance, surveillance, data acquisition, military applications and much more. Its manoeuvrability also makes them appealing for indoors usage where MAVs need to handle narrow corridors and acute turns.

Control model of physics of the quad-rotor is a system of four identical propellers aligned in square pattern. Each of these propellers generates thrust and torque along normal axis of the square. It is assumed that the thrust is controlled directly and that the torque is directly proportional to the respective thrust.

2.2 Swarm

The control model for escape behaviour presented in [4] is based on BOIDS model [2] that is inspired by natural behaviour of large groups of certain animals like schools of fish, swarms of insect or flocks of birds. In this thesis, this model is implemented as distributed algorithm for each MAV without communication with other members of the swarm.

2.3 Acting forces

The forces influencing MAVs can be divided in three groups.

- **Other individuals effect** – this is sum of forces between a certain individual and all other members of the swarm. This force keeps the swarm together and also prevents collisions in the swarm.
- **Goal effect** – this is the force that pulls each member of the swarm towards its destination, effectively moving whole swarm forward.
- **Obstacle effect** – this force pulls MAVs away from obstacles that are too close.

Resulting force acting on each individual is composed as weighted sum of all these forces.

3 Adjustments to the original algorithm

3.1 Dynamic goal

First thing that comes to mind when speaking about navigating through a complex environment is the ability to follow a given path. In [4], such ability was not implemented. It simply placed one stationary target to the final destination. And because of the stationary goal, it was also necessary to change the equation used to compute the force pulling the swarm towards the goal by adding an extra condition and limit result force in order to prevent it from being too large and thus overpowering other two forces.

Here a different approach was chosen and moving target was implemented. This target moves in front of the swarm in a predefined constant distance along the path and thereby leads the swarm through the environment since the target follows the path given to the swarm. More ways of positioning the target were tested and the one that places the target ahead of the perpendicular projection of the center of mass of the swarm was chosen due to its best performance.

However, this approach had insufficient performance when the path turned as can be seen in Figure 2 where the first two MAVs fly needlessly towards the second obstacle. This flaw was proportionally more obvious with more sharp turns and even more relevant with bigger swarms. To solve this problem it was decided that one target is not enough and each MAV was given its personal target to chase. This improvement applied on the same situation can be seen in Figure 3.

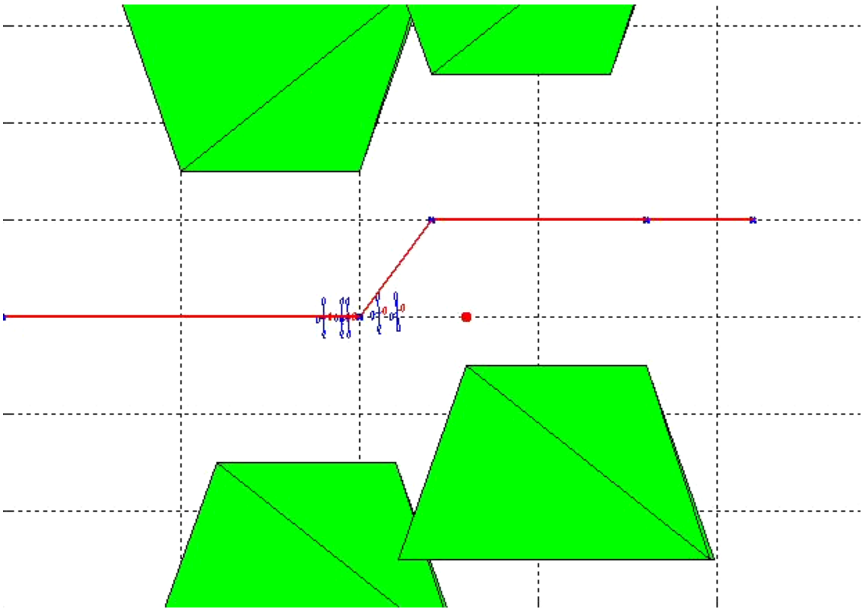

Figure 2: Example of one shared target for whole swarm.

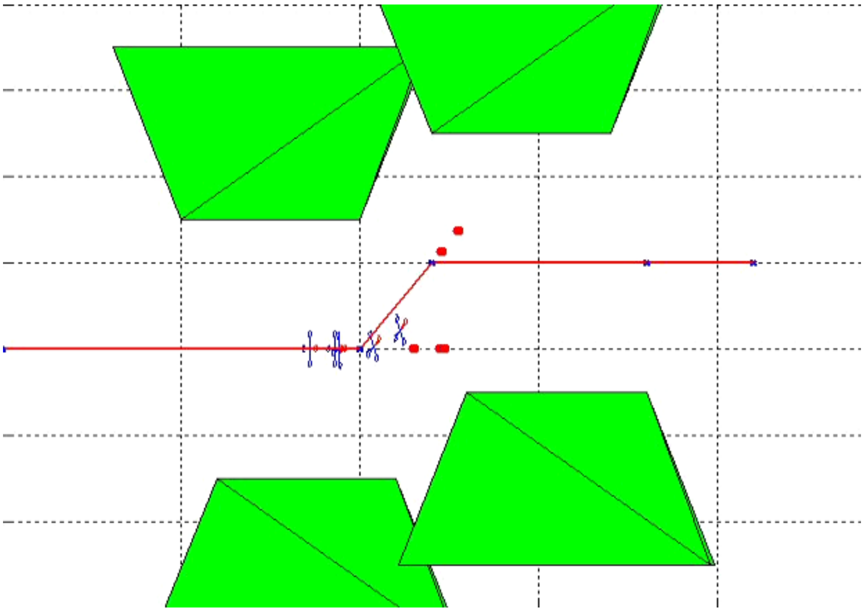

Figure 3: Example of personal target for each MAV.

Such individual target is always on axis defined by current section of the path in fixed distance ahead of the perpendicular projection of the particular quad-rotor. This allows the swarm to follow any given path closely and therefore this approach satisfies the requirements for precise navigation of the swarm indoors.

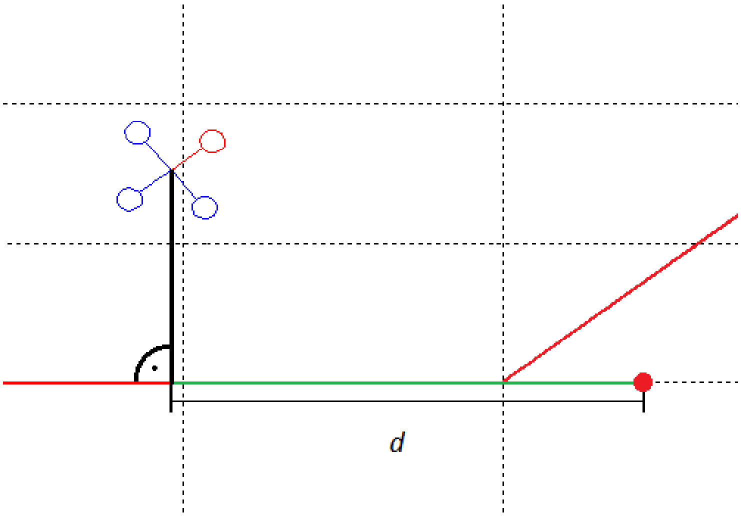

Figure 4: Scheme of target placement. The center of the MAV is projected perpendicularly to the path it follows (red line) and the target (the red dot) is placed in constant distance *d* (green line) ahead.

3.2 Polyhedral obstacles

As mentioned in the beginning, obstacles were limited to constant size spheres in [4]. It was necessary to implement means to define more shapes of obstacles in order to simulate complex environments and get closer to real world application.

At first, a solution of how to implement polyhedra into the algorithm that would be iterative and user friendly was needed. Since the GJK algorithm works with convex shapes only, it is satisfactory to make a list of vertices of any arbitrary polyhedron and use the inbuilt function of MATLAB [6] to make the necessary computations of which vertices are actually connected by edges. This makes the process of adding new obstacles easy since the GJK algorithm uses vertices too.

3.3 The GJK algorithm

Most of the information in this chapter comes from [7, 8] where the GJK algorithm is described.

The GJK algorithm was first published in 1988 and is named after his inventors - scientists Gilbert, Johnson and Keerthi. Since then it went through various developments and depending on the actual implementation it is capable of detecting collisions, computing minimal distance between two objects or computing the depth of an intersection. Here the version that can compute minimal distance between two objects is used.

The main advantages of this algorithm are:

- Speed - it is iterative algorithm but it converges extremely fast, in experiments presented in this thesis it was usually less than five iterations.
- Robust on convex shapes – it needs convex shapes with so called "support function" (more in chapter 3.3.4) implemented. But then it can run on any combination of these convex shapes without problems and not care whether it is polyhedron or curved shape which is major problem in other algorithms.

More reasoning for using this algorithm can be found in [5]. In the following paragraphs the GJK algorithm will be explained for the case computing the distance between objects.

3.3.1 Convex shape

Convex shape is a shape where any two points of this shape can be connected with straight line and every point of this line will still be inside the object (see Figure 5).

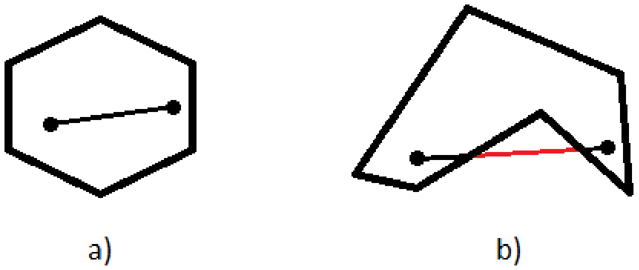

Figure 5: Convex shape vs. concave shape

3.3.2 Minkowski sum

The GJK algorithm relies heavily on a concept called the Minkowski sum. The Minkowski sum is a concept where from two shapes a third shape is created by taking every single point in one shape and add them to every point in the second shape like

$$A + B = \{a + b | a \in A, b \in B\}. \tag{1}$$

Important thing of this concept is that when a Minkowski sum of two convex shapes is computed, the result will always be a convex shape too.

To implement this algorithm, subtracting points of one shape from another instead of adding them is necessary. Sometimes, this can be found named as the "Minkowski difference" but "Minkowski difference" does not exist, it is still Minkowski sum even with subtraction operator:

$$A - B = \{a - b | a \in A, b \in B\}. \tag{2}$$

The reason for computing this Minkowski sum is that when the two shapes are intersecting, the resulting shape will contain origin. That is because if two objects are intersecting, some points will have the same coordinates and thus when subtracted from each other, they will project into the origin. Moreover, and for this thesis more importantly, if they are not intersecting, the distance between the origin and the resulting shape is the same as the distance between the two original shapes.

When working with convex shapes, all that is necessary is to compute just the difference of vertices of each of those shapes and the final shape will be complete. This would greatly reduce the number of computations and speed up the algorithm but for the GJK algorithm it is not necessary to compute the whole Minkowski sum.

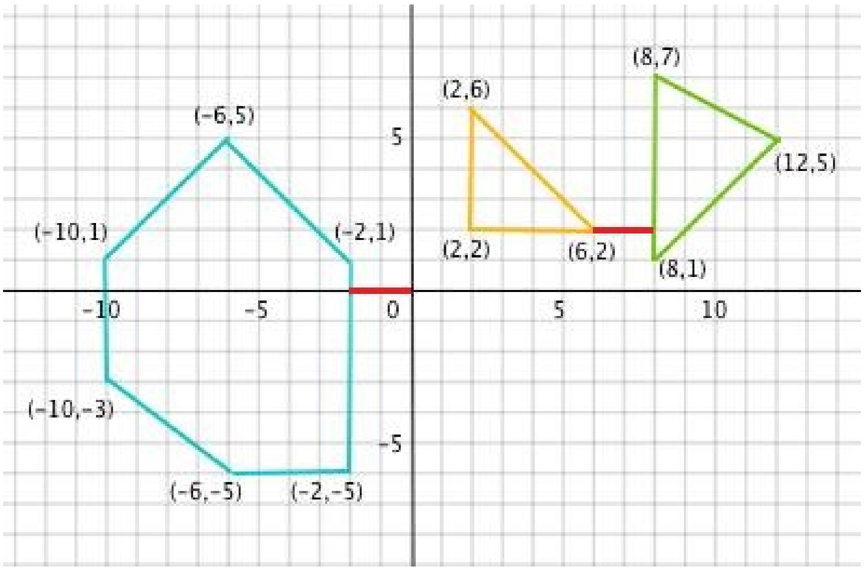

Figure 6: Two original shapes (the triangles on the right side) and resulting Minkowski sum created with subtraction operator, both with highlighted distance vectors

3.3.3 The Simplex

It is sufficient to find which part of the Minkowski sum is closest to the origin and compute the distance between that part and the origin. An object called the simplex is built inside the Minkowski sum to do this. In geometry, a k-simplex is a k-dimensional polytope which is the convex hull of its k + 1 vertices.

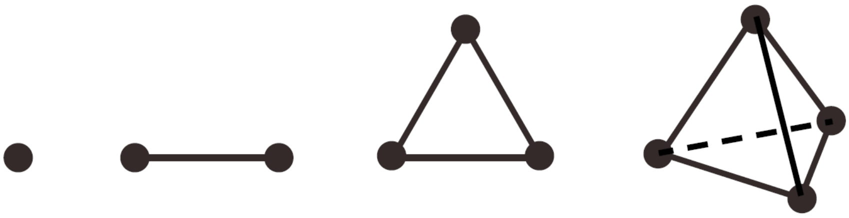

Figure 7: Simplices from 0-simplex to 3-simplex [11]

3.3.4 The support function

The last thing that has to be explained before introducing the algorithm itself is the support function which was mentioned at the beginning. This function will be called on all objects appearing in the GJK algorithm, namely obstacles and quad-rotors, and it is different for each type of object. This function returns the point of the object that is farthest along the direction defined by the main loop (see chapter 3.3.5). This thesis uses only polyhedra to define objects so this function will return the vertex of the polyhedron that returns the highest value of dot product with the direction vector. This makes the process of adding new type of object into the system really easy because it is only required to create a support function for that new type of object.

3.3.5 Main loop

Here the algorithm will be explained step-by-step with pseudocode following (see scenario 1).

1. To initialize the algorithm a simplex with the same amount of vertices as dimensions of the space is needed. For flying machines it will therefore be a triangle. The support function is used to create this simplex.
    - First a random direction is chosen and a point farthest along that direction of the first object and farthest in the opposite direction (the Minkowski sum uses subtraction in this algorithm) of the second object are selected and added together. This is the first point of the simplex.
    - Then the same process is made in the opposite direction to find the second point and lastly in a direction perpendicular to the previous directions to find the third point. These three points together create the initial simplex with vertices overlapping the vertices of the Minkowski sum.
2. The next step is to pick a normal vector of the simplex in the direction to origin and try to search for a point of the Minkowski sum that is farther along this direction than the simplex. If the point found this way is on the plane specified by the simplex, then the simplex is already as close to the origin as possible and the distance between the simplex and the origin is computed.
3. If the support function found a point that is farther along the direction to the origin, this point will replace the point in the simplex that is the farthest from the origin itself. Then the algorithm continues with the next iteration of the loop from step 2.

```
Input: $A, B$ – convex objects whose distance will be computed
$d \leftarrow$ pick any arbitrary direction (in this thesis the vector from the MAV
    to the first vertex of the obstacle is chosen);
$Simplex \leftarrow support(A, B, d)$   // add point in direction $d$ to the $Simplex$;
$Simplex \leftarrow support(A, B, -d)$;
$\bar{d} \leftarrow$ perpendicular to $d$;
$Simplex \leftarrow support(A, B, \bar{d})$;
while true do
    $d \leftarrow$ normal to the simplex in direction to origin;
    $p \leftarrow$ support(A, B, d);
    if $p$ is not farther along $d$ than Simplex then
        return point from Simplex closest to origin;
    else
        $Simplex \rightarrow$ remove point farthest from origin;
        $Simplex \leftarrow p$
```

Scenario 1: GJK algorithm main loop outline

4 Real world restrictions

In this chapter, another part of the resulting system will be discussed, which is integration of the constraints given by using real MAVs.

4.1 Collisions between quad-rotors

One of the problems of [4] is that it does not prevent quad-rotors from collisions with neighbours in the swarm reliably. There is a force that pulls MAVs apart when they are too close to each other but this force is linear and so it can be easily overpowered by other acting forces.

In order to prevent this situation, it was necessary to change this force from linear to exponential and re-stabilize the swarm. Therefore, a new function with exponential ascend in the beginning was developed, which prevents MAVs from colliding reliably. In the original system, the resulting force was computed as a multiple of linear force and corresponding weight of the force $e_{i,j}$. This system keeps the linear force the same and changes the weight function.

The new weight function is divided in two parts depending on the actual distance between the quad-rotors and each part is computed with different equation. The first part that is used for distances $< l^1$ is described as

$$e_{ij} = \frac{d}{L_{ij} - size} - l, \tag{3}$$

where L_{ij} is the distance between individuals i and j. The second part that is used for distances $\geq l$, is described by

$$e_{ij} = \frac{1}{e^{a*L_{ij}-b} + c} + \frac{1}{e^{0.5*a*L_{ij}-b} + c}, \tag{4}$$

and is the same as in [4]. The resulting weight function can be seen in Figure 8.

This set-up has satisfactory performance on all tested scenarios although the swarm is noticeably more volatile and can be difficult to balance because of the added exponential force.

4.2 Air flow influence

Another problem coming from the real world is the fact that rotor-based flying machines generate downward air flow which provides lift for the machine (see Figure 9). This creates

[1] The variables mentioned in this part were set after numerous experiments as following in this thesis: $a = 6$, $b = 6$, $c = 0.6$, $d = 2$, $l = 1.4$, $size = 0.4$.

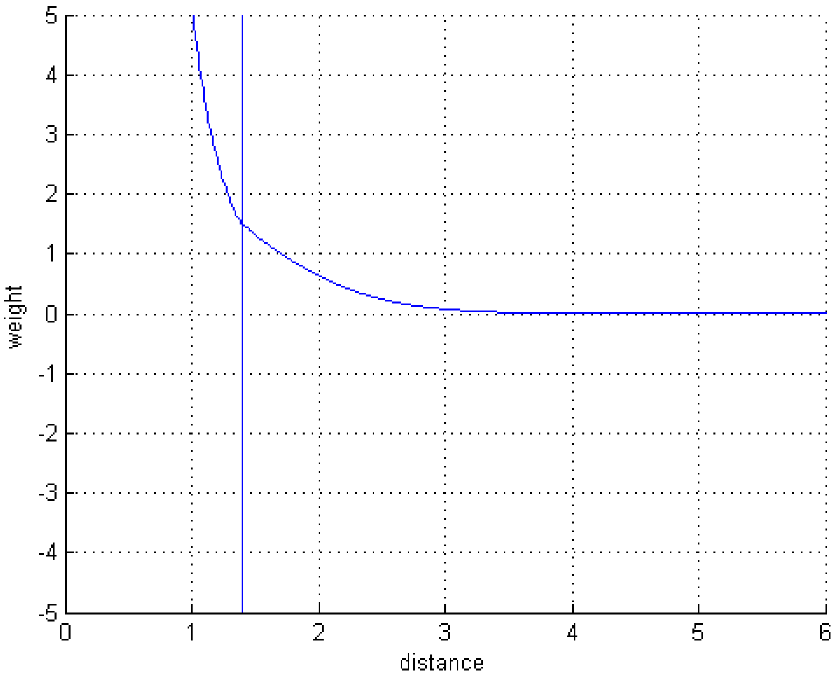

Figure 8: Weight function for force acting between individuals with highlighted boundary between equations (3) and (4).

no problems if a machine flies alone but when dealing with swarms, it is necessary to keep them from flying above/below each other. If the airflow generated by the quad-rotor with higher altitude hits the one below, the lower MAV would lose its lift and would inevitably fall to the ground.

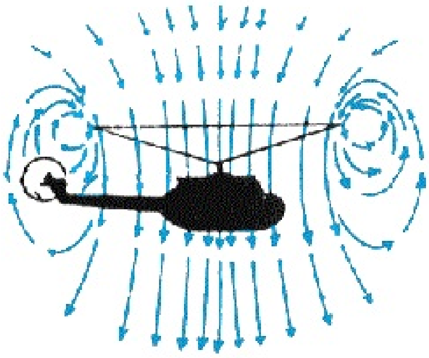

Figure 9: Air-flow induced by rotor-craft. Source: www.dynamicflight.com

Because of this restriction, another force was added into the system to keep all the quad-rotors in the swarm out of vertical space belonging to another quad-rotor. For the same reasons as in chapter 4.1 this force is designed as exponential one. It is the same function as in Figure 8, only its diameter is reduced (see Figure 10) for reasons described in chapter 4.3.

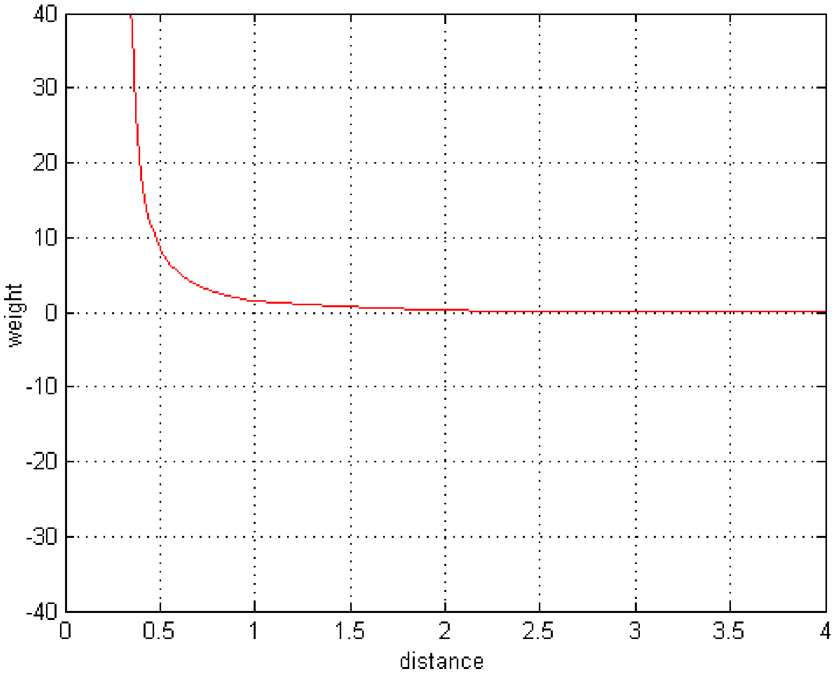

Figure 10: Weight function for horizontal force between individuals.

4.3 Summary

Both of these forces are designed and can be adjusted independently but because they are closely tied together and act in accord with each other, they were developed simultaneously. The main reason to have both of these forces is to keep the swarm compact while discouraging the MAVs from flying in perfectly even plane without reducing this problem to 2D only. Either of these forces could easily keep the swarm together but they both have their flaws:

- The force described in chapter 4.1 does not take into account the airflow influence. This approach, although with linear weight, was used in [4].
- The force described in chapter 4.2 reduces the problem into 2D. It is able to keep the swarm compact and takes into account the airflow influence but it omits the vertical

dimension completely. An obstacle could move an individual up or down from the swarm and it would not be noticed.

Because of these reasons, this thesis uses both of the previously mentioned forces to form a swarm that uses three dimensional space fully. Example of how these forces form a swarm shows Figure 11.

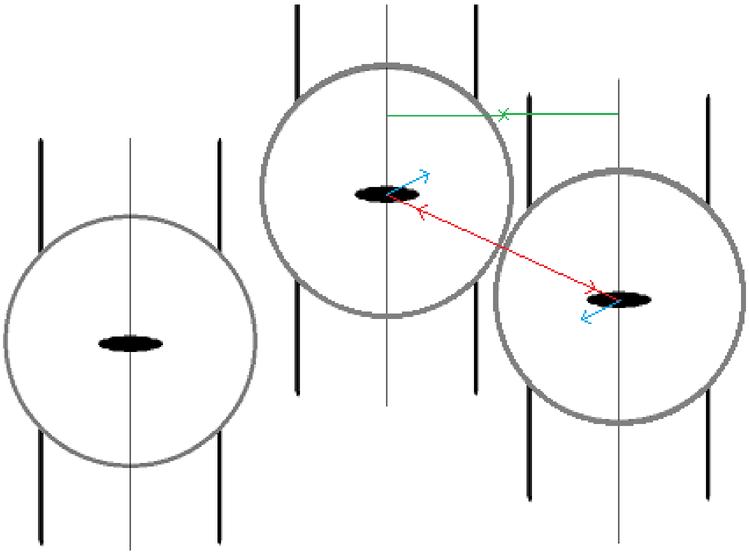

Figure 11: Scheme of a swarm with highlighted forces acting between two individuals. Green are forces described in chapter 4.2 (here attractive), red are forces between individuals from chapter 4.1 (here repulsive) and blue is the final force.

Last of the limitations influencing behaviour of the MAVs is the top speed limit. This is important for more reasons. First, because real MAV is usually equipped with camera facing down that keeps measuring the speed of flight and it can keep up with the speed only until certain point. Also the changes of the forces described in this chapter and simulations in narrow environments described in chapter 6 generates situations where multiple strong forces are acting at once. Hence it was necessary to ensure that the MAVs will keep their speed on a safe level. This was achieved by limiting the total force acting on individual.

5 Relative localization

The aim of this part of the system is to keep the swarm compact based on relative localization of other members. Each member of the swarm keeps paying attention to other quad-rotors and checks that at least three other members (two if the swarm is composed of six members or less) of the swarm are closer than a certain boundary[2].

If this condition fails, another force needs to come into play and pull the quad-rotor closer to its neighbours that began to be lost so that it will maintain distance that allows their relative localization, and therefore keep the swarm compact

$$e_{ij} = -\frac{d}{L_{ij} - boundary} - k, \tag{5}$$

where L_{ij} is the distance between the individuals. This force is designed as exponential one with slower ascend than the forces keeping the quad-rotors from collisions but still powerful enough to pull the swarm together. Because of its close relation with forces from the previous chapter, mainly from chapter 4.1, this force is not designed independently. Instead it changes part of the force shown on Figure 8 to pull the distant MAVs together. Changed weight function for this force is on Figure 12.

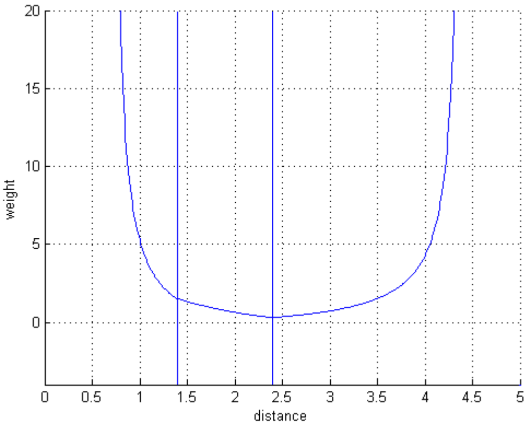

Figure 12: Changed weight function for force between individuals in case of loosing sight with highlighted boundaries between equations (3), (4) and (5).

Several simulations were run to verify that the relative localization works as intended and can be seen in chapter 6.2.

[2] The variables mentioned in this part were set after numerous experiments as following in this thesis: $d = 2$, $k = 0.7$, $boundary = 4$

6 Experiments

6.1 Passing through an alley

One of the main goals of this thesis was to evaluate behaviour of the swarm in an environment that puts a lot of stress on the swarm – a narrow alley. The alley can be represented by any gap narrow enough to squeeze the swarm from both sides. Horizontal gaps are not excluded by default but this thesis deals with vertical gaps for two reasons:

1. The swarm is usually spread widely but on vertical axis all the members are mostly even. This is result of using rotor-crafts that are unable to fly above each other. This problem is addressed in chapter 4.2. Because of this, horizontal gaps do not change the shape of the swarm.
2. Vertical gaps are more common than horizontal ones, as an example we can mention doors.

An example of basic alley used in these experiments can be seen in Figure 13 but other scenarios were tested too to provide more accurate results. All of the alleys are listed in Table 1 together with short descriptions of their shape.

Id	Name	Definition
0	*no alley*	*empty space*
1	Basic alley	default
2	Long alley	an alley of at least double length
3	Uneven alley	an alley with an uneven entry
4	Two alleys	two alleys in row with enough open space between them
5	Wide entry alley	an alley that has wider entry than its final width
6	S-shaped alley	an alley with two turns, right and left in quick succession

Table 1: Summary comparison of various alleys used in experiments described in chapter 6.1.

Numerous simulations were run and different parameters were adjusted to find common settings that would stabilize the swarm in as many different scenarios as possible. Summaries of simulations with these settings are listed below in Figures 15-22.

These experiments showed that it is very difficult to estimate the delay caused by passing through an alley. There is always delay but it is not substantial one. Moreover the results vary and the same scenarios have different results for different sizes of the swarm. There were even three occasions for the swarm of 27 MAVs where the swarm was quicker by

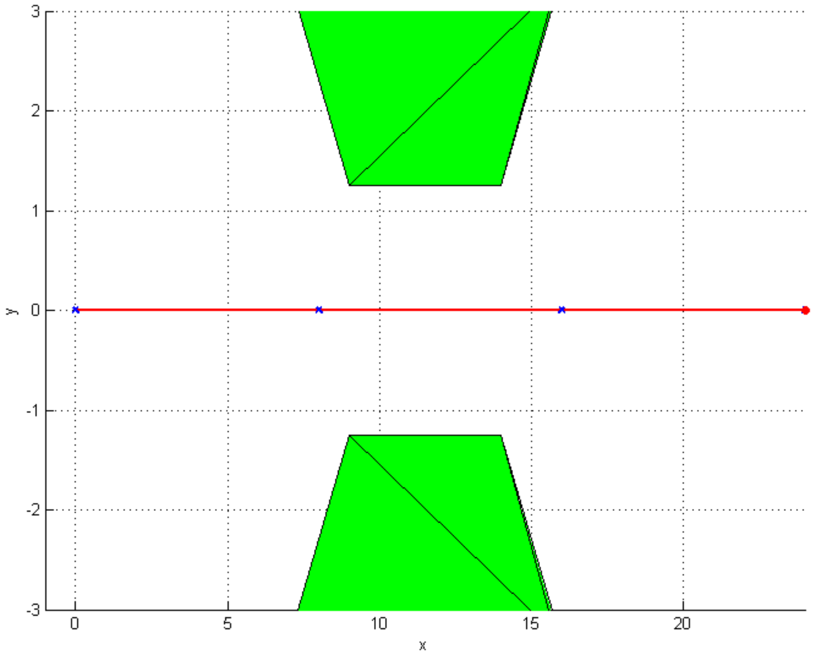

Figure 13: Scheme of the basic alley (red line is the path that the swarm would follow).

about 0.1 second for two seconds at the entrance to an alley in comparison to free flight as shows Figure 22. This unusual behaviour is caused by the fact that swarm of this size slows itself down naturally because the individuals at edges are not pulled straight forward by the goal force but to the center and are pushed away by other individuals that are closer to the center.

The most significant effect on the final delay is the exact location of each MAV. As this cannot be foretold in any way, it is not possible to estimate the delay precisely and it is required to run simulation for specific environment and swarm to analyse the behaviour of the swarm (which is necessary to plan the route for the swarm).

On the other hand, it is possible to define minimum width of an alley for any swarm of known size, that the system presented in this thesis with its specific settings is able to clear. This relation is shown in Figure 14 and appears to be proportional.

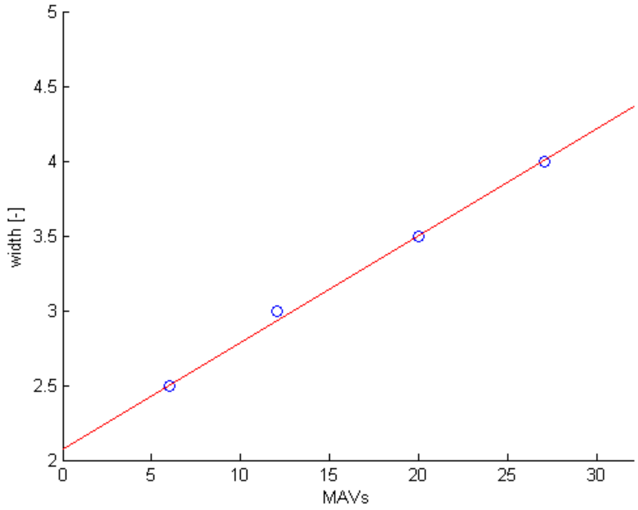

Figure 14: Graph of dependency of minimum width of a basic alley on the size of a swarm. Blue are measured values and red is the line fitted to these data.

6.1.1 Results for 6 MAV

Here ale listed results for a swarm of 6 MAVs in different alleys.

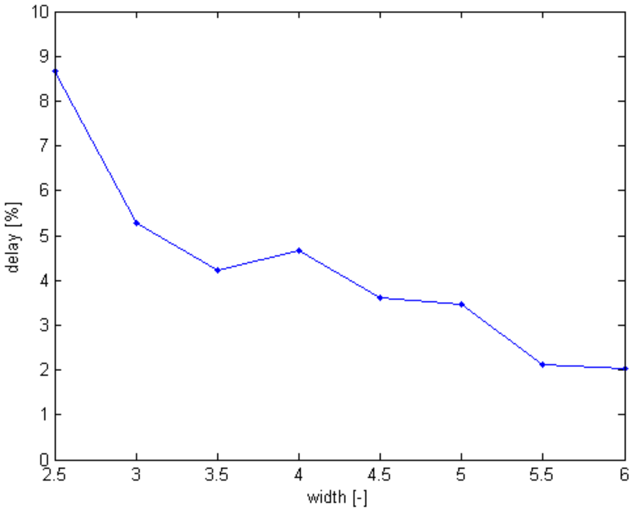

Figure 15: Relative comparison of the same swarm passing through the basic alley with various width showing slowdown percentage.

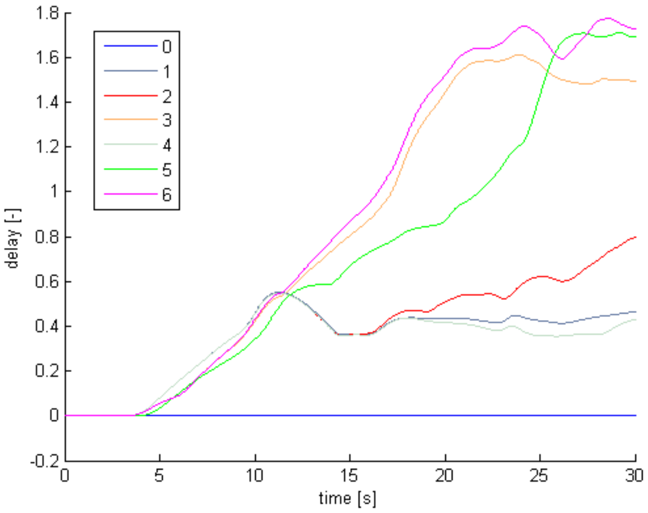

Figure 16: Relative comparison of the slowdown of the same swarm passing through different shaped alleys (see table 1 for description of the IDs of alleys) with constant width of 3 and the free flight as a baseline.

6.1.2 Results for 12 MAV

Here ale listed results for a swarm of 12 MAVs in different alleys.

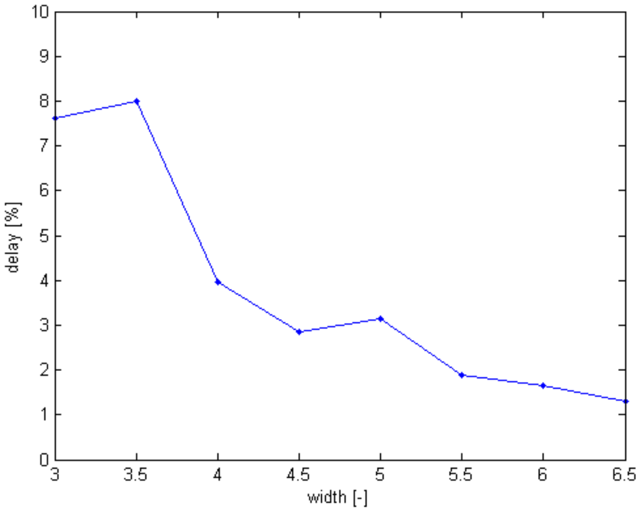

Figure 17: Relative comparison of the same swarm passing through the basic alley with various width showing slowdown percentage.

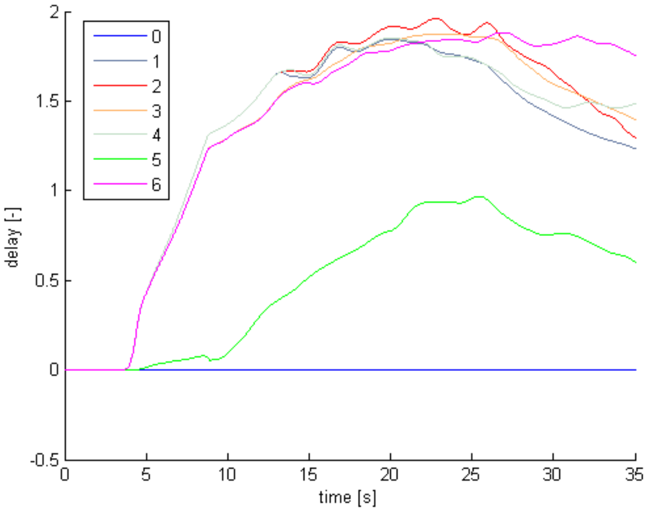

Figure 18: Relative comparison of the slowdown of the same swarm passing through different shaped alleys (see table 1 for description of the IDs of alleys) with constant width of 3.5 and the free flight as a baseline.

6.1.3 Results for 20 MAV

Here ale listed results for a swarm of 20 MAVs in different alleys.

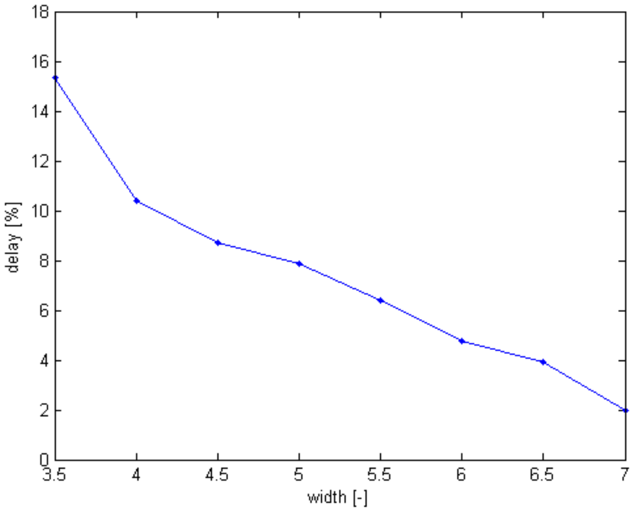

Figure 19: Relative comparison of the same swarm passing through the basic alley with various width showing slowdown percentage.

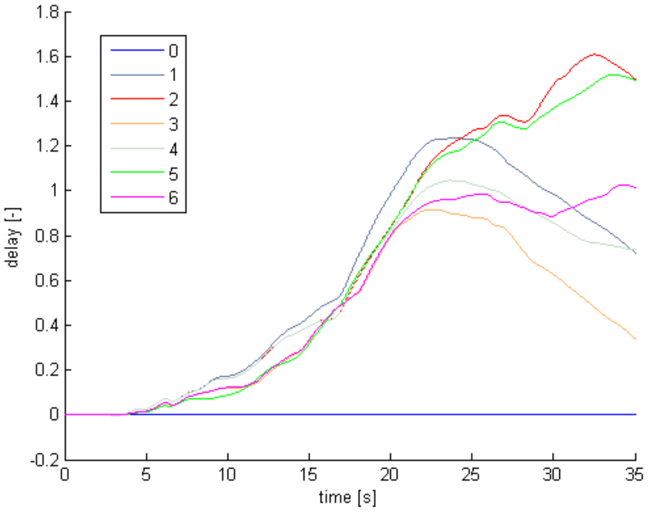

Figure 20: Relative comparison of the slowdown of the same swarm passing through different shaped alleys (see table 1 for description of the IDs of alleys) with constant width of 4 and the free flight as a baseline.

6.1.4 Results for 27 MAV

Here ale listed results for a swarm of 27 MAVs in different alleys.

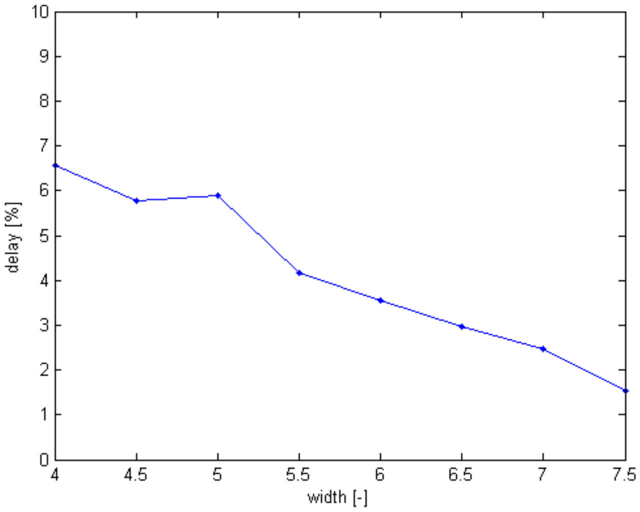

Figure 21: Relative comparison of the same swarm passing through the basic alley with various width showing slowdown percentage.

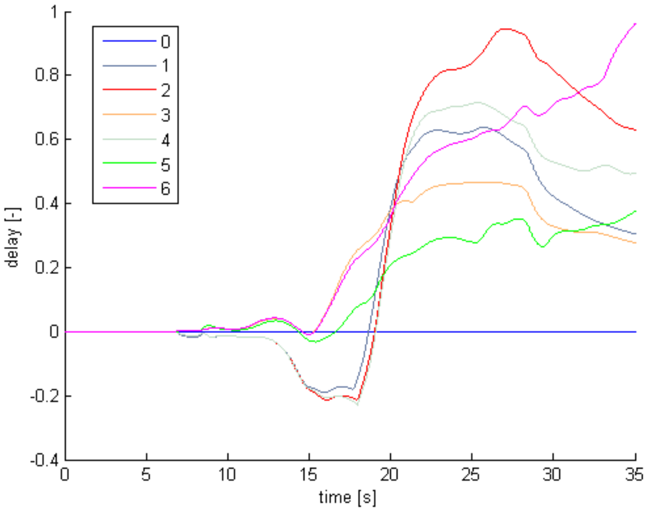

Figure 22: Relative comparison of the slowdown of the same swarm passing through different shaped alleys (see table 1 for description of the IDs of alleys) with constant width of 4.5 and the free flight as a baseline.

6.2 Relative localization evaluation

Here are shown examples to demonstrate that the forces described in chapter 5 keep the swarm in a distance that allows their relative localization in all cases (see Figures 23-30). There are three lines for each MAV that show the distance to the three closest neighbours and two red lines which represent the borders in which the MAVs are encouraged to stay. It can be seen that in some cases when dealing with big swarms, few lines go below the bottom border for some time like in Figure 29. This means that they get closer than their comfortable zone but there is enough space reserve around MAVs to prevent any collision. It is because of the size of the swarm, that they can get closer than usual before such a big swarm gets stabilized.

6.2.1 Swarm of 6 MAVs

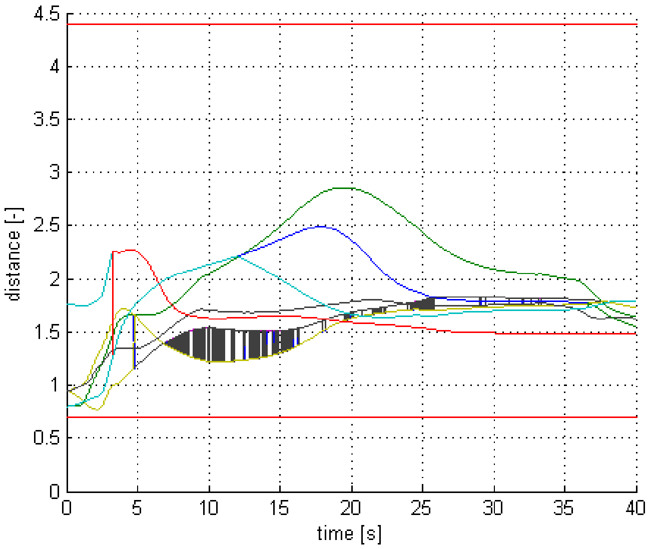

Figure 23: The three shortest distances to its neighbours for each MAV in open space.

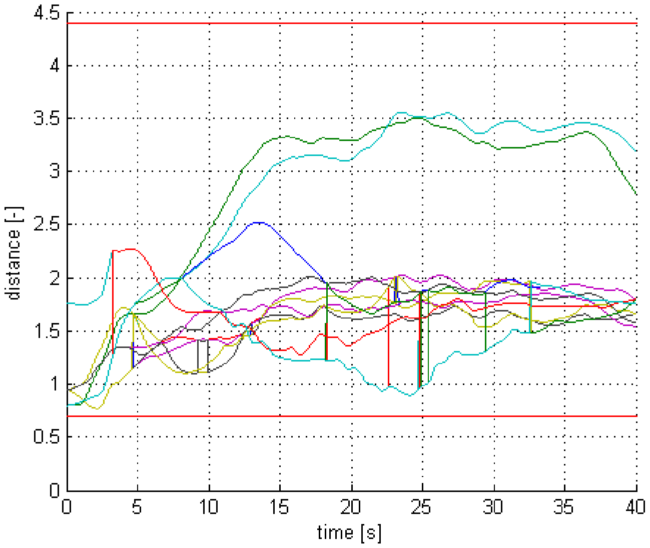

Figure 24: The three shortest distances to its neighbours for each MAV in S-shaped alley of width 3.

6.2.2 Swarm of 12 MAVs

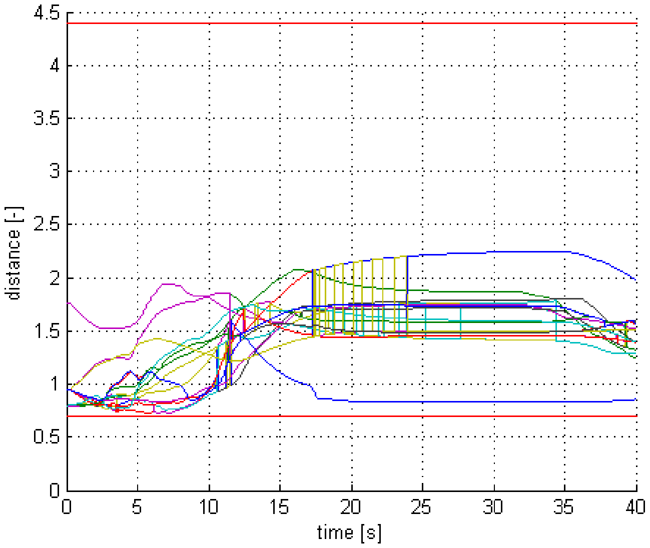

Figure 25: The three shortest distances to its neighbours for each MAV in open space.

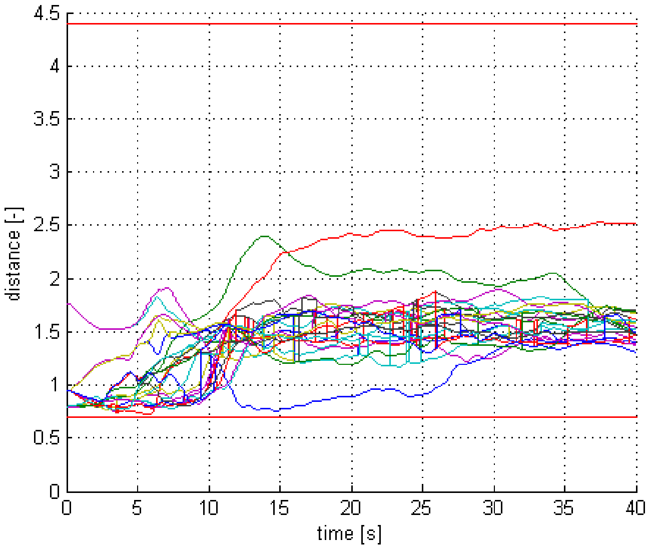

Figure 26: The three shortest distances to its neighbours for each MAV in S-shaped alley of width 3.5.

6.2.3 Swarm of 20 MAVs

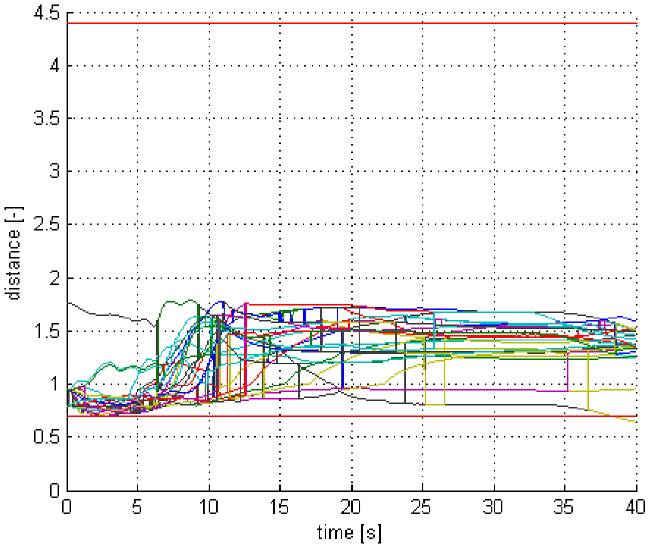

Figure 27: The three shortest distances to its neighbours for each MAV in open space.

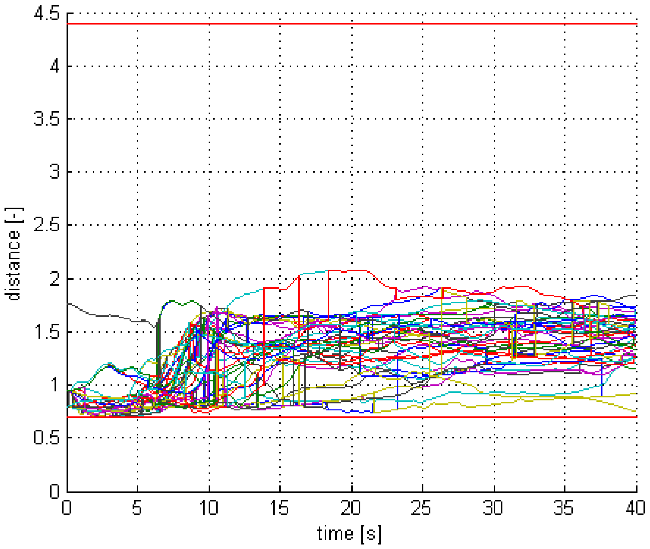

Figure 28: The three shortest distances to its neighbours for each MAV in S-shaped alley of width 4.

6.2.4 Swarm of 27 MAVs

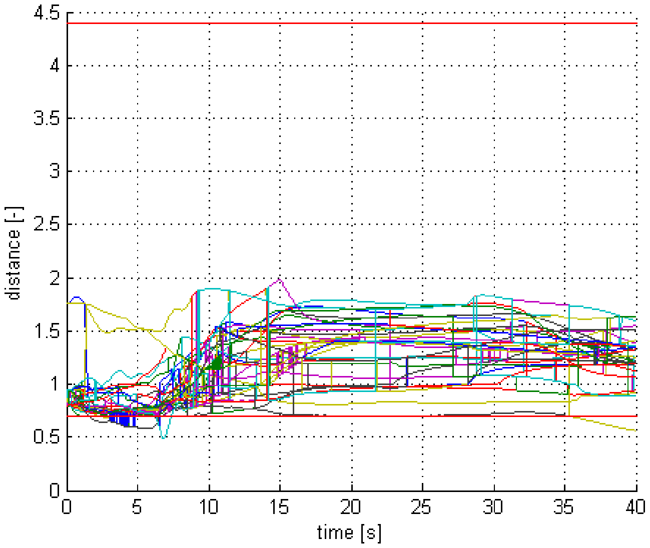

Figure 29: The three shortest distances to its neighbours for each MAV in open space.

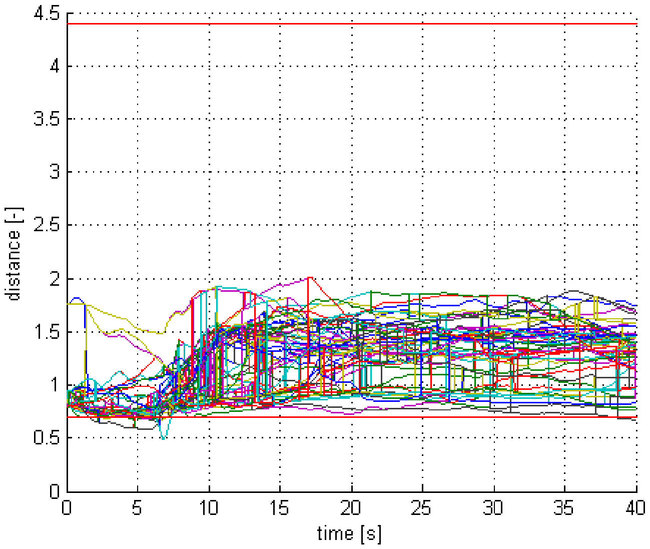

Figure 30: The three shortest distances to its neighbours for each MAV in S-shaped alley of width 4.5.

6.3 Indoors environment

Other experimental environments created to evaluate the algorithm are examples of complex environments. One is an outline of hallway and kitchen area (see Figure 31). Another set of experiments was run with big bars obstacle (Figure 34). Third complex environment evaluated in this section is a corridor filled with different obstacles, one following after another (outline in Figure 37).

6.3.1 Kitchen area simulations

This is a good example of indoors area (Figure 31), it features kitchen cabinets in a small room and hallway with two doors used as checkpoints and serving as tight passages to put stress on the swarm. Because of the size of MAVs, this cramped indoor simulation was run with only 3-6 members (see example in Figure 32).

There were measured only little differences (up to about 6 seconds delay which is $\sim 10\%$ increase) between swarms in terms of total flight time (Figure 33).

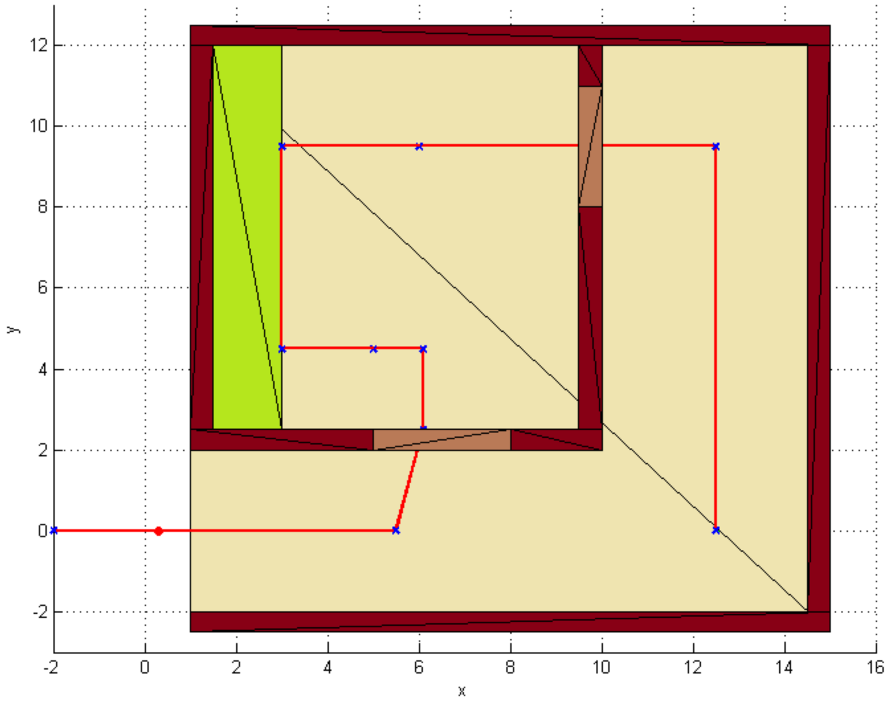

Figure 31: Scheme of the indoors kitchen-like environment featuring small rooms, multiple doors and kitchen cabinets.

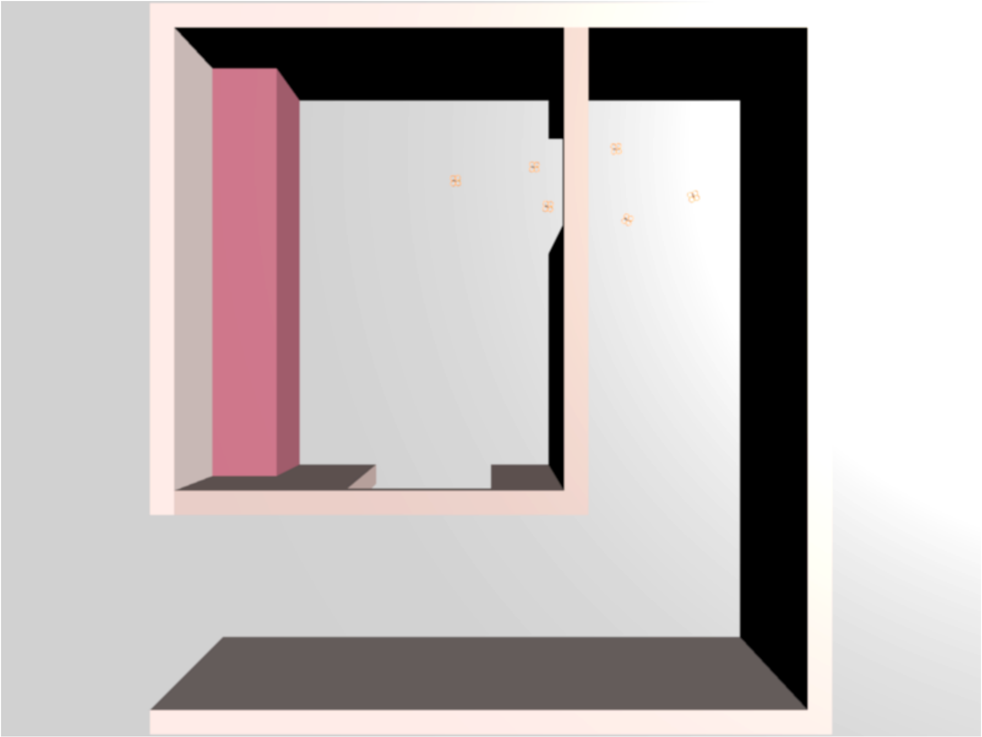

Figure 32: Snapshot of the swarm of 6 MAVs flying through the kitchen area.

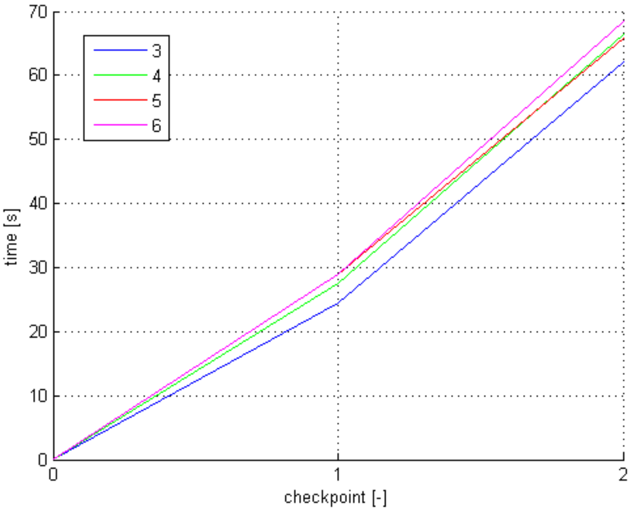

Figure 33: Total time needed by different sized swarms to clear checkpoints (doors) in the kitchen area.

6.3.2 Big bars simulations

This model situation is here to demonstrate another difficult type of obstacle that the swarm can take on. Example of the set-up of this obstacle is shown on Figure 34.

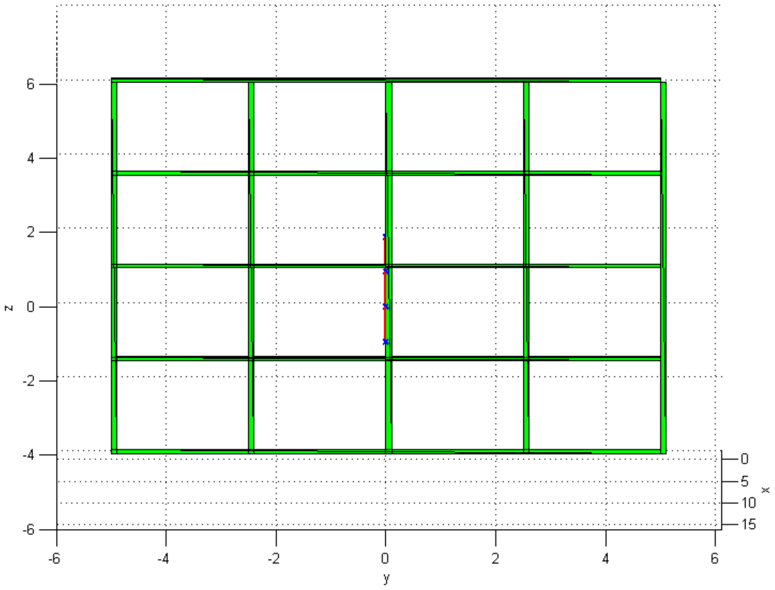

Figure 34: Scheme of the big bars obstacle

This obstacle caused by far the biggest problems for the swarm from all complex environment simulations run in this thesis. First set-up had one more bar in the grid for the same size (i.e. gaps were squares only 2x2 wide instead of current 2.5x2.5) but the swarm was not able to pass through. There were cases where some MAV in the end managed to clear that obstacle but mostly it was similar to navigating the swarm against a wall.

The gaps were widened only by 25% to keep bar-like type of obstacle and it was sufficient even for bigger swarms (see Figure 35). Results for different swarms can be seen in Figure 36.

This experiment could easily end with different outcome if the bars would have been moved a bit, if they were placed diagonally or if they were cylindrical. It has much more unpredictable outcomes than the alleys described in chapter 6.1 and the conclusion is the same, the most significant effect is the exact location of each MAV.

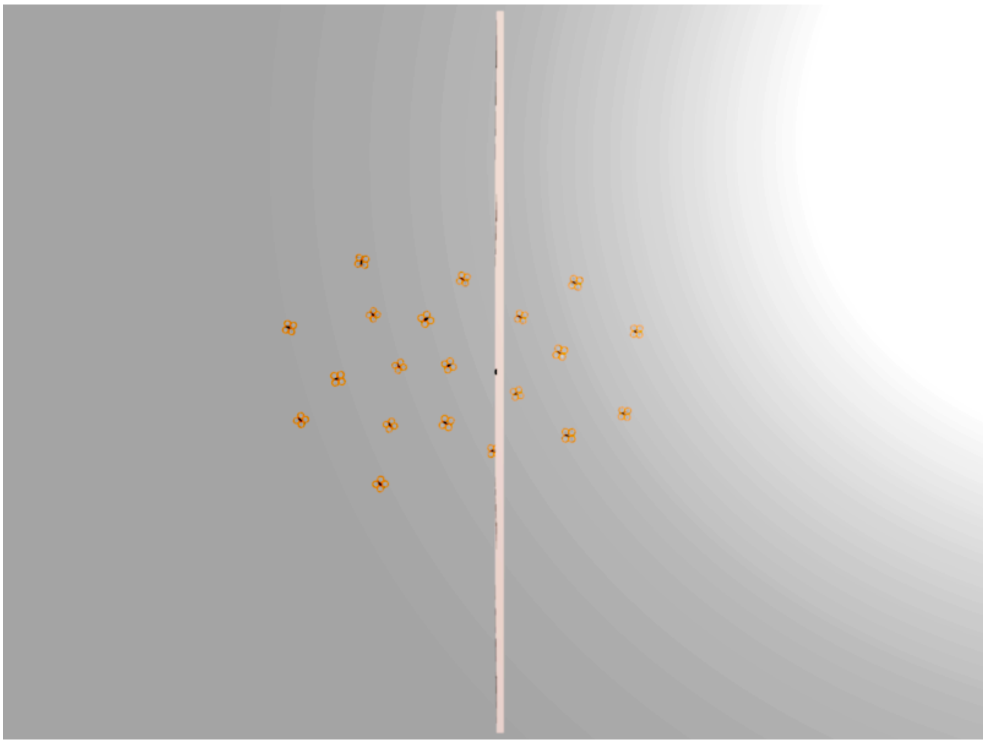

Figure 35: Snapshot of the swarm of 20 MAVs flying throug the bar obstacle.

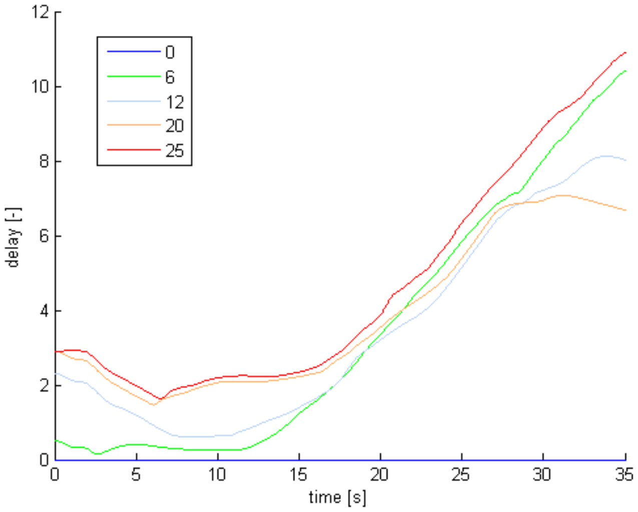

Figure 36: Comparison of the delay on the bars obstacle for various sized swarms

6.3.3 Maze-like corridor simulations

Yet other example of complex environment is a short maze-like corridor (Figure 37) with following obstacles: a horizontal bar sticking out from the right wall right in the middle of the path for the swarm, a window in a wall, a perpendicular wall leaving only half of the space for the swarm and finally two pyramid corners sticking out from the walls one after another and the second turned upside down.

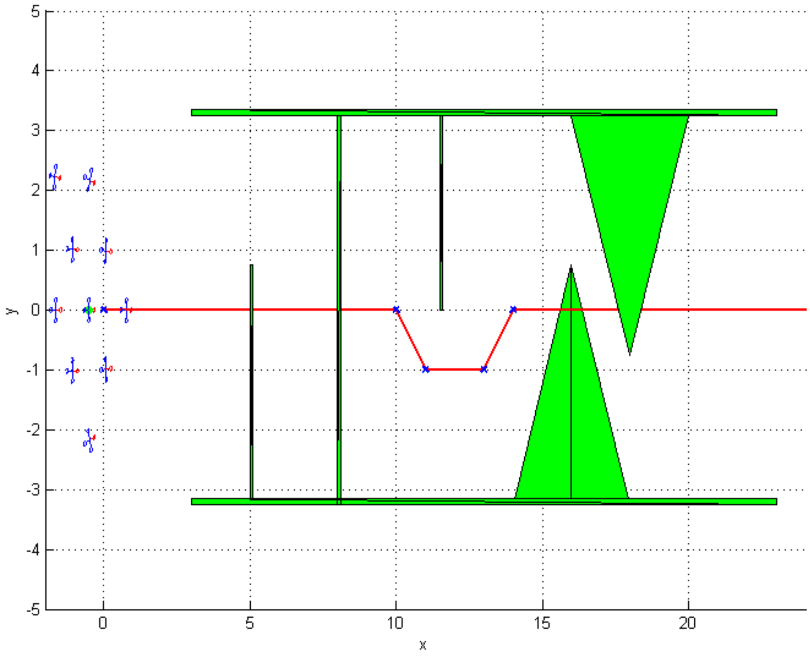

Figure 37: A scheme of the maze-like corridor

This represents wide variety of obstacles standing in the way of the swarm and the swarm needs to avoid these obstacles without precise navigation around them. The only exception is the wall perpendicular to the course of the swarm where the swarm needs a little help, that means to move the path from the wall. Without this change the left side of the swarm would be navigated straight into the wall. Figures 38-40 show how the swarm handles these obstacles. Figure 41 demonstrates the differences and similarities between swarms consisting of 6 MAVs and 20 MAVs respectively. See the results in Figure 42.

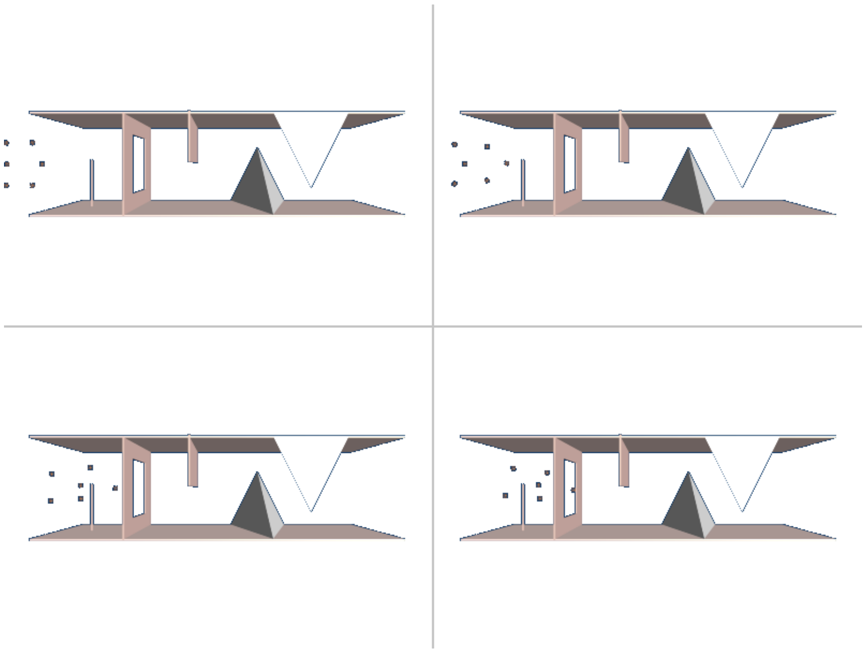

Figure 38: Snapshots of the swarm of 6 MAVs flying through the maze-like corridor.

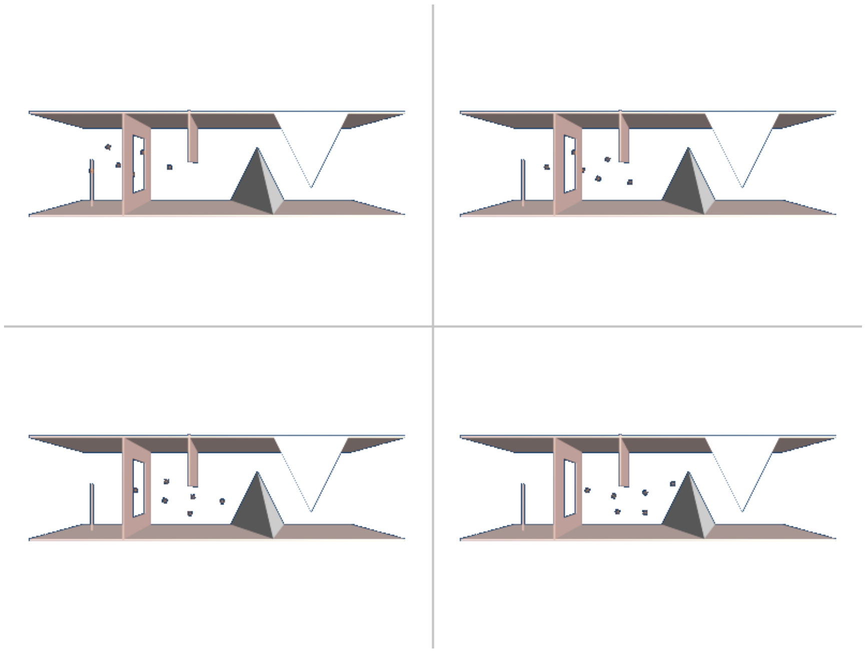

Figure 39: Snapshots of the swarm of 6 MAVs flying through the maze-like corridor.

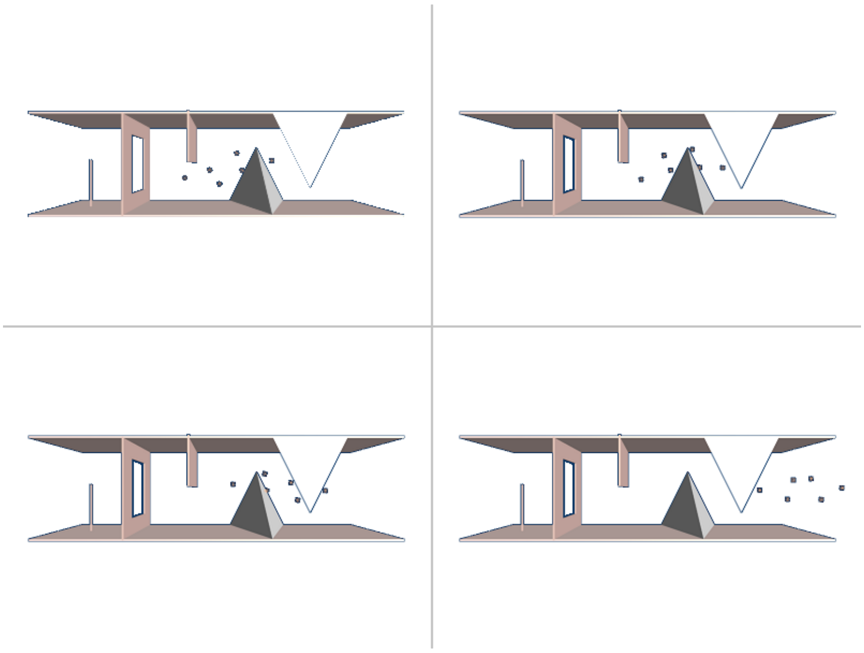

Figure 40: Snapshots of the swarm of 6 MAVs flying through the maze-like corridor.

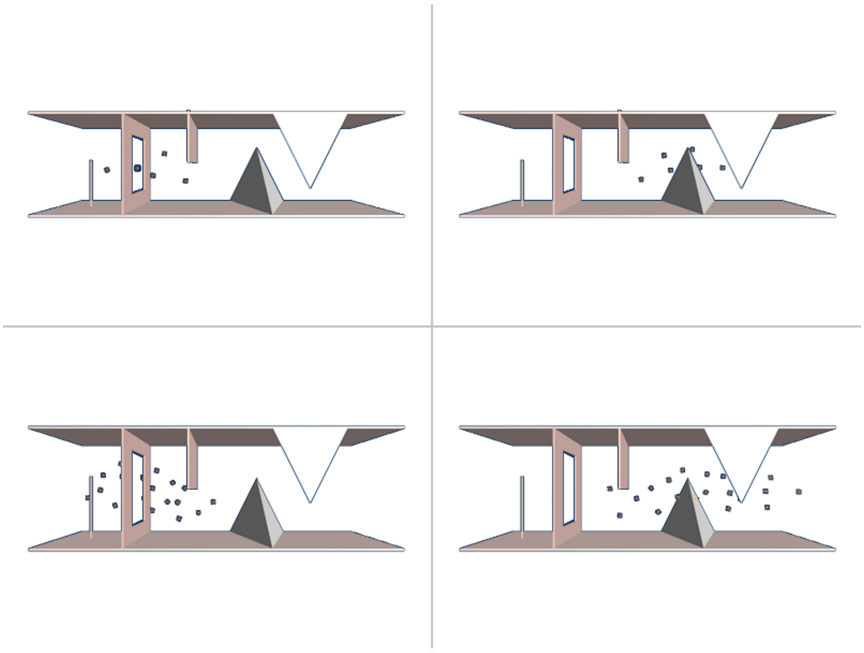

Figure 41: Comparison of two different sized swarms. First row displays swarm of 6 MAVs, second swarm of 20 MAVs.

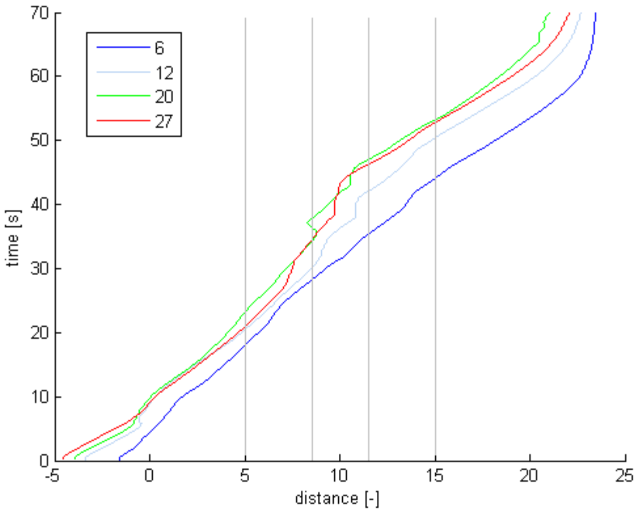

Figure 42: Total time needed to clear the maze for different sized swarms

7 Conclusion

The robust system for navigating the swarm of MAVs through a complex environment was successfully developed. The core of this system comes from [4] and is enhanced to provide more variability, stability and functionality.

An ability of following a predefined path was implemented by using an approach with a moving target. A lot of different approaches were tested and in the end an individual receding target for each MAV was implemented. The target is placed directly on the path in a constant distance ahead the MAV as described in chapter 3.1.

The support for simulating the swarm behaviour in a complex environment was added by using convex polyhedra to form obstacles and a new algorithm to detect these obstacles, the GJK algorithm, which was implemented in the system. This is described in chapter 3.

Numerous changes were made during the attempts to stabilize the swarm while implementing constraints given by real MAVs. The results of experimental evaluation are described in chapter 4. These modifications changed the behaviour of the swarm towards safety. Collisions between MAVs were eliminated in all tested cases and collisions with obstacles were removed as well. This system, however, is not intended to ensure safety when navigated wrongly, for instance when navigating big swarms in alleys too narrow or into the wall. On these occasions some MAV can hit an obstacle or collide with another.

Relative localization functionality was implemented and now every MAV is aware of its neighbours and keeps itself in their proximity. This is the topic of chapter 5.

And finally in chapter 6, the whole system was tested in different scenarios with many different settings. Results were analysed and rules for passing through different alleys were formulated in chapter 6.1. The relative localization restrictions were met in all tested cases without trouble as can be seen in chapter 6.2. Also various different complex scenarios were created and multiple simulations were run to verify robustness of the system.

To sum up, all objectives of the thesis assignment were fulfilled:

- The ability to follow a dynamic target receding ahead the swarm along a given path was designed and implemented.
- The GJK algorithm was implemented to allow successful navigation of the swarm through a complex environment.
- Constraints given by using real helicopters were added to the system.
- Rules for visual relative localization were implemented.
- There were made many simulations in numerous alley-like scenarios as well as in complex environments and the results were analysed.

The possible continuation of this work could be in advanced distribution of each group of forces that are described in chapter 2.3. Now the distribution is constant in all cases which means for example that the swarm is slowing down to regroup even in open spaces where it is not needed. The analyses made in chapter 6.1 could be used to predict moments where the swarm would get into trouble and split the swarm into several smaller swarms. Also automated planning algorithm could be implemented to find the path instead of manually setting one.

Relevant information on MAV control and planning achieved by other members of Multi-robot Systems group can be found in [9, 10] and on web *mrs.felk.cvut.cz*.

References

[1] Saska, M. - Vakula, J. - Přeučil, L.: *Swarms of micro aerial vehicles stabilized under a visual relative localization*, Proc. of IEEE International Conference on Robotics and Automation (ICRA), 2014.

[2] Reynolds, C. W.: *Flocks, herds, and schools: A distributed behavioral model, in Computer Graphics*, 1987, pp. 25–34.

[3] Faigl, J. - Krajník, T. - Chudoba, J. - Přeučil, L. - Saska, M.: *Low-Cost Embedded System for Relative Localization in Robotic Swarms* Proc. of IEEE International Conference on Robotics and Automation (ICRA), 2013.

[4] Vakula, J.: *Escape behavior in swarms of unmanned helicopters*, 2012.

[5] Sekanina, T.: *Algorithms for navigation of swarm of unmanned helicopters and following a dynamic target in a complex environment*, 2014.

[6] MathWorks Documentation, "convhulln"
http://www.mathworks.com/help/matlab/ref/convhulln.html, 2015.

[7] Muratori, C.: videolecture "Implementing GJK"
http://mollyrocket.com/849, 2006.

[8] Bittle, W.: "GJK (Gilbert-Johnson-Keerthi)"
http://www.codezealot.org/archives/88, 2010.

[9] Saska, M. - Vonasek, V. - Krajnik, T. - Preucil, L.: *Coordination and Navigation of Heterogeneous MAV-UGV Formations Localized by a 'hawk-eye'-like Approach Under a Model Predictive Control Scheme*, International Journal of Robotics Research, 2014.

[10] Saska, M. - Chudoba, J. - Preucil, L. - Thomas, J. - Loianno, G. - Tresnak, A. - Vonasek, V. - Kumar, V.: *Autonomous Deployment of Swarms of Micro-Aerial Vehicles in Cooperative Surveillance*, Proceedings of 2014 International Conference on Unmanned Aircraft Systems (ICUAS), 2014.

[11] Palmieri - Antieau - Scrimshaw - Stump - King: "Finite simplicial complexes"
http://www.sagemath.org/doc/reference/homology/sage/homology/simplicial_complex.html, 2009.

Appendix

CD Content

In table 2 are listed names of all root directories on CD

Directory name	Description
thesis	bachelor thesis sources in latex format
sources	MATLAB source codes
animations	animations recorded during development
figures	all figures created during development

Table 2: CD Content