3D Robot Formations Planning with Fast Marching Square

David Álvarez, Javier V. Gómez, Santiago Garrido and Luis Moreno.
Robotics Lab, Department of Systems and Automation.
University Carlos III of Madrid, Leganés, 28911, Spain.
E-mail: {dasanche, jvgomez, sgarrido, moreno}@ing.uc3m.es

***Abstract*— This research presents a novel approach for 3D robot formation motion planning. The methodology presented is based on the standard Fast Marching Square (FM^2) path planning method and its application to robot formations motion planning. For the formation coordination, a leader-followers scheme is used, which means that the reference pose for the follower robots is defined by geometric equations that place the goal pose of each follower as a function of the leader's pose. The use of the Frenet-Serret frame in order to control the orientation of the formation is introduced. Thanks to the combination of these methods, the configuration of the formation is able to adapt its shape depending on the environment conditions. This adaptation is based on the velocities map calculated in the first step of the FM^2 algorithm. Also, robot priorities within the formations are introduced. This is an important contribution since it provides different behaviours to the formation members in special situations. Using this information, simulations show that the method is able to achieve good performance in difficult environments.**

I. INTRODUCTION

In the recent years, research on multi-robot coordination systems in 3D environments has increased exponentially due to the price drop of unmanned aerial vehicles (UAV) and the advent of micro-air vehicles (MAV) as a popular robotic test bed. Besides, in many tasks the use of multi-agent systems increases in overall mission performance, flexibility and robustness without augmenting the capacity of each UAV unit [1]. These characteristics can be applied to many different areas of research such as: exploration [2], search and rescue [3], surveillance [4], [5], and many others. In order to achieve a good performance in any of these applications several research topics need to be addressed, such as: modelling and control of such agents [6], collision avoidance [7], mapping and state estimation with such agents [8] or formation control and planning [9].

In formation control, a group of coordinated robots have to perform a specific task trying to keep a certain geometric configuration. There are many issues to be considered when designing a controller for mobile robot formation, such as the stability of the formation, the controllability of formation patterns, safety or uncertainties in formations. All these problems need to be addressed while the formation is moving along different scenarios. Therefore the configuration of the formation is likely to evolve over time while the formation is performing the task due to space or safety constraints.

In order to control the way the formation changes, many strategies have been used. In [10] the multi-agent coordination problem is studied under the framework of control Lyapunov functions. The main idea is that every robot has a control Lyapunov function and that there exists a control Lyapunov function for the formation of robots, which is a weighted sum of individual control Lyapunov function of each robot. The main drawback is the mathematical complexity introduced in order to obtain satisfactory results. Other works use an approach based on potential fields which are combined in order to get the desired behaviour of the formation [11]. A major problem when applying potential fields to the planning problem is the existence of local minima. In behaviour-based approaches [12] each robot has basic primitive actions that generate the desired behaviour in response to sensory input. Possible schemas include collision avoidance and goal seeking. The virtual structure introduced by [13] is defined as a collection of agents that maintain a desired geometric configuration. The algorithm has three main steps: the virtual structure is aligned with the position of the robots, then a trajectory for every agent is obtained using mission control, and finally each robot follows its own path. This approach is capable of maintaining a highly precise formation and has mainly used for satellite formation control [14]. However due to its high computational complexity is very difficult to apply it to multi-UAV control.

In this research the Leader-Followers (LF) approach has been used. In this strategy, a robot (that could be virtual) is designated as the leader of the formation and follows a trajectory towards the goal point while pulling the followers behind it according to a pre-defined geometry specification that can change within a given range in order to accommodate to the environment conditions [15], [16], [17]. An advantage of this strategy lies on its facility of implementation since no feedback loop from followers to the leader is needed. Another important characteristic is that it depends on the leaders motion, so it is very important to have a very good path planning and tracking, because once the leader loses its path, its error is fully propagated to all the followers and both the mission and coordination objectives can fail.

In this paper, we focus on the control of the robot formation in a 3D environment using a LF architecture. The leader's path is calculated using the Fast Marching Square (FM^2) path planning method, which ensures obtaining very safe and smooth paths [18]. For the followers, the pre-defined geometry evolves dynamically, while the leader is covering the path, based on a velocities map which is calculated as a first step of FM^2. This map provides a very easy way to deal with robot priorities when going through very narrow environments, and also an easy way to take into account dynamic obstacles when updating the followers positions. Besides, the algorithm used for the formation to evolve has a very low mathematical complexity, making it very easy to apply and change when different behaviours are desired.

The next sections of the paper are organized as follows. Section II introduces the Fast Marching Square path planning method. In section III the appliance of FM^2 to the robot formation problem is explained. In section IV different simulation results are shown. Finally, in section V conclusions and future work are addressed.

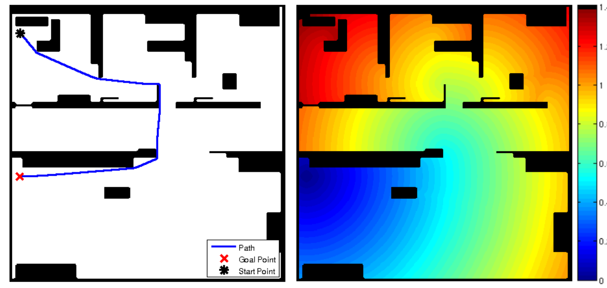

Fig. 2: Example of a path obtained with the FMM. The left side shows the original map and the path calculated. In the right there is the map of distances computed with FMM.

II. Fast Marching Square path planning method

A. The Fast Marching Method

If we consider a light ray travelling through different materials, the Fermat's principle says that its path is the one that consumes a minimum time. This is especially interesting in our application, because if we have only a source of light waves (goal point for the path), each point in the space is connected with the source with a path that it is parametrized by the time of arrival of the wave. Besides, considering the set of all the points of the domain with the time as last coordinate, we can create a Lyapunov surface in which the level curves are isochronals and the Fermat's paths are orthogonal to them. This means that it is impossible for the method to have local minima. Graphically, this can be seen in figure 1, which represents the funnel potential of the light wave propagation with a constant refraction index.

From the point of view of path planning, these considerations lead us to think that computing the same path for robots can lead to a high save of time. Mathematically, the propagation of the light is given by the Eikonal equation. In [19] an approximation of the solution for this equation was proposed, the Fast Marching Method (FMM). Let us assume a 2D map, where $\mathbf{x} = (x, y)$ is a point on the map with the coordinates in relation to a Cartesian referential, $D(\mathbf{x})$ is the front wave arrival time function and $W(\mathbf{x})$ is the velocity of the wave propagation. Besides, we assume that a wave starts propagating at time $D = 0$ with velocity W always

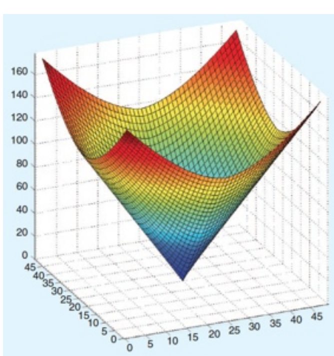

Fig. 1: Lyapunov surface when the propagation wave starts in a point and the refraction index is constant.

non-negative. The Eikonal equation (1) defines the time of arrival of the propagating front wave, $D(\mathbf{x})$, at each point $\mathbf{x}$, in which the propagation speed depends on the point, $W(\mathbf{x})$, according to:

$$|\nabla D(\mathbf{x})| W(\mathbf{x}) = 1 \tag{1}$$

Discretizing the gradient ∇D according to [20] it is possible to solve the Eikonal equation at each point $p(x_i, y_j)$, where i and j are the row and column of a grid map, as follows:

$$\begin{aligned} D_1 &= min(D_{i-1,j}, D_{i+1,j}) \\ D_2 &= min(D_{i,j-1}, D_{i,j+1}) \end{aligned} \tag{2}$$

$$\left(\frac{D_{i,j} - D_1}{\triangle x}\right)^2 + \left(\frac{D_{i,j} - D_2}{\triangle y}\right)^2 = \frac{1}{W_{i,j}^2} \tag{3}$$

The FMM consists on solving $D_{i,j}$ for every point of the map starting at the source point of the wave where $D_{i_0,j_0} = 0$. The following iterations solve the value $D(i, j)$ for the neighbours of the points solved in the previous one. Using as an input a binary grid map, the output of the algorithm is similar to the distance transform, but in this case is continuous, not discrete. These distances have the meaning of the time of arrival of the expanding wave at every point in the map. After applying the FMM, gradient descent can be used from any point of the map of distances to obtain a path towards the source of the wave, which works as a goal point. This is valid only if one wave has been employed to generate the map of distances. The main advantage of this method is that the path obtained is optimal in distance, like the example in figure 2.

B. Fast Marching Square Method

As we can see in figure 2, although optimal in distance, it is obvious that the path produced by FMM is not safe in terms of distance to the obstacles, nor feasible in terms of the abruptness of the turns that the path requires. These problems lead us to consider using the Fast Marching Square (FM^2) method as path planner. The FM^2 [21] solves these two main disadvantages. It is based on applying the FMM twice. After

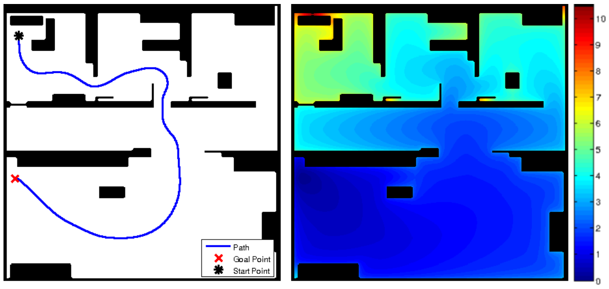

Fig. 3: Example of a path obtained with the FM^2. The left side shows the resulting path of the algorithm in the initial map. In the right side there is the time of arrival map.

the first time it is applied, the resulting map of distances is considered as a map of velocities for the second FMM step. This means that the second time the wave is propagated, the velocity at which it moves can be different at every point in the map. Besides, this velocity is proportional to the distance to the closest obstacle, meaning that the wave is faster when it is far from obstacles. This produces important differences in the path that is computed, as it can be seen in figure 3.

The proposed FM^2 algorithm has several properties which make it very good for path planning purposes [16], [17]. Most important ones include:

- *No local minima*: as long as only one wave is employed to generate the map of distances, FMM ensures that there is a single global minimum at the source point of the wave (goal of the path).
- *Completeness*: the method finds a path if it exists and notifies in case of no feasible path.
- *Smooth trajectories*: the planner is able to provide a smooth motion plan which can be executed by the robot motion controller. In other words, the plan does not need to be refined.
- *Reliable trajectories*: it provides safe (in terms of distance to obstacles) and reliable trajectory (free from local traps). This avoids the coordination problem between the local collision avoidance controllers and the global planners, when local traps or blocked trajectories exist in the environment.
- *Fast response*: if the environment is static, the map of velocities is calculated only once. Since the FMM can be implemented with a complexity order of $O(n)$ [22], building the map of velocities is a fast process.

C. 3-Dimensional Fast Marching Square

Since the FM^2 algorithm is based on the standard FMM, it is extensible to more than 2 dimensions. This is the case of this paper, since it is applied to 3D robot formations planning. The algorithm works exactly in the same way as the 2D version, with the only difference that the front wave becomes a spatial curve. Also, the time response is a little slower since the size of the grid that models the environment is much bigger. Despite of this, all the properties of the FM^2 remain in a n-dimensional environment. This is the main fact that lead us to use this algorithm as path planner.

III. Robot Formation planning with FM^2

The algorithm described next is an adaptation of the one proposed in [23], [24], in which the FM^2 path planning method is used to control 2D formation in different scenarios.

In this research, the leader-followers scheme is used for robot formation path planning. The reference pose for the follower robots is defined by geometric equations that place the goal pose of each follower as a function of the leader's pose. The leader can be a robot, a person or even a virtual leader. The path for the leader is computed in a egotistic way, not taking into account the formation. The FM^2 provides a two-level artificial potential which repels the robot from obstacles, but working only with this artificial potential could lead to the robots crashing into each other. Thus, robot formation motion control requires additional repulsive forces between robots. Integrating the potential given by FM^2 and the repulsive force between robots, each robot has at each moment one single potential attracting it into the objective but repelling it from obstacles and other robots. The main requirement when integrating all the potentials is to do it in a way that does not create local minima. Figure 4 shows the steps of the algorithm on a triangle-shaped robot formation. Although it is a 2D shape, it has been chosen because it is easier to understand the behaviour of the followers.

A. Robot pose coordination

In a 3D formation a problem arises when dealing with orientation of the followers. Since the reference for the formation geometry is the leader's trajectory, we need to define some vectors in this trajectory so that we can place the follower robots. Meanwhile in 2 dimensional space normal and tangent vectors are used, as shown in figure 4, the same concept cannot be applied in 3 dimensions since it turns out that there exist an infinite number of perpendicular vectors to the tangent to the trajectory. All of these vectors are contained in a plane, as depicted in figure 5.

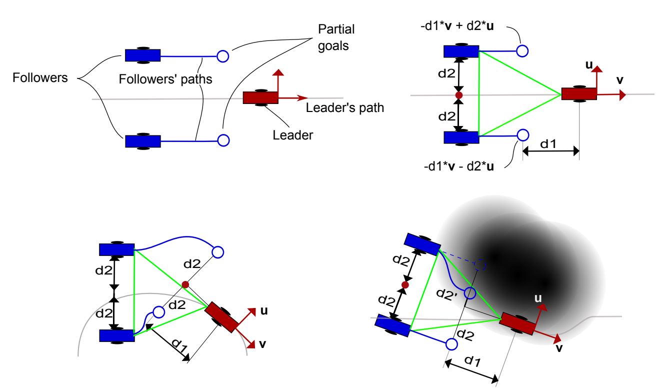

Fig. 4: Top left - Main components of the robot formation algorithm. Top right - Reference geometric definition of a simple, triangle-shaped robot formation. bottom left - Behaviour of the partial goals depending on the leader's pose. Bottom right - Behaviour of the partial goals depending on the obstacles of the environment.

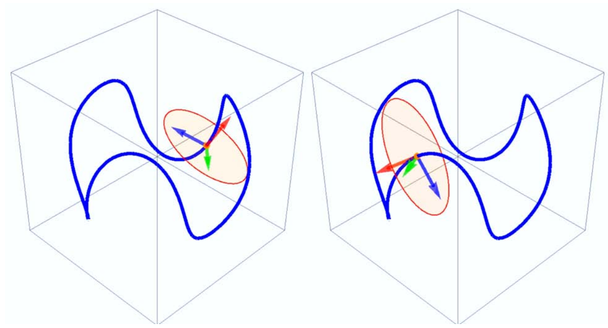

Fig. 5: The red vector represents the tangent to the trajectory and the red circle the perpendicular plane to this vector.

In order to adapt the proposed robot formation planning method to 3 dimensions, a third reference has to be defined so that the reference geometry of the formation can be defined. In our case, a relative reference based on the Frenet-Serret formulae [25], [26] is used. It extracts the local characteristics of the path as a third reference, this way it allows the formation to be environment-independent when defining the reference geometry.

The Frenet-Serret formulae are used to describe the kinematic properties (velocity, curvature and torsion) of a particle which moves in a three-dimensional Euclidean space, $\mathbb{R}^3$. Let $\mathbf{r}(t)$ be a parametrization of a continuous, differentiable curve C in a Euclidean space $\mathbb{R}^3$. Let us denote $\mathbf{T}(t)$, $\mathbf{N}(t)$ and $\mathbf{B}(t)$ as the unit tangent vector, unit normal vector and unit binormal vector respectively. Let us denote also the curvature as $\kappa(t)$ and the torsion as $\tau(t)$, the Frenet-Serret formulae are:

$$\begin{cases} \mathbf{T}'(t) = \kappa(t)|\mathbf{r}'(t)|\mathbf{N}(t) \\ \mathbf{N}'(t) = -\kappa(t)|\mathbf{r}'(t)|\mathbf{T}(t) + \tau(t)|\mathbf{r}'(t)|\mathbf{B}(t) \\ \mathbf{B}'(t) = -\tau(t)|\mathbf{r}'(t)|\mathbf{N}(t) \end{cases} \quad (4)$$

The Frenet-Serret frame is defined by the collection of the three vector functions $\mathbf{T}(t)$, $\mathbf{N}(t)$ and $\mathbf{B}(t)$ satisfying the following fundamental relations:

$$\mathbf{T}(t) = \frac{\mathbf{r}'(t)}{|\mathbf{r}'(t)|} \quad \mathbf{N}(t) = \frac{\mathbf{T}'(t)}{|\mathbf{T}'(t)|} \quad \mathbf{B}(t) = \mathbf{T}(t) \times \mathbf{N}(t) \quad (5)$$

The graphical representation of the Frenet trihedron is already shown in Figure 5, where the red vector is $\mathbf{T}(t)$, the blue vector is $\mathbf{N}(t)$ and the green vector represents $\mathbf{B}(t)$. The advantage of using the Frenet trihedron is that among the infinite possible vectors perpendicular to the tangent vector, one is chosen in the direction of the curvature $\mathbf{N}$ (or normal acceleration). Furthermore, the direction of this vector changes continuously which is an important property when applying this trihedron as a reference for the geometry formation. Then, by combining vectors $\mathbf{T}$, $\mathbf{N}$ and $\mathbf{B}$ any shape can be given to the formation as depicted in figure 6.

B. Formation planning algorithm

The FM^2 uses a two-step potential to compute the path: the first step creates a potential which can be interpreted

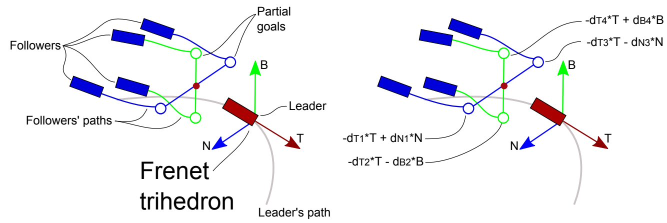

Fig. 6: Schema of the definition of the geometry in 3D based on the Frenet trihedron.

as a velocities potential, which we denoted as $W(\mathbf{x})$; and the second step creates a funnel shaped potential, which represents the distance to the goal in the metrics $W(\mathbf{x})$ and is denoted as $D(\mathbf{x})$. The robot formation path planning algorithm using FM^2 is the following:

- The environment map $\mathbf{W_0}$ is read as a binary map, where 0 (black) means obstacles or walls and 1 (white) means free space. This map is common for all the robots in the formation (both leaders and followers).
- The first potential $\mathbf{W}$ is calculated applying the FMM to the binary map $\mathbf{W_0}$, according to the FMM 1st step of the FM^2.
- The second potential $\mathbf{D}$ is calculated applying the FMM on the potential $\mathbf{W}$.
- An initial path for the leader is calculated applying gradient descent on the potential $\mathbf{D}$, according to the FM^2 method.
- Once we have a path for the leader of the formation, a loop, in which each cycle represents a step of the robots' movement, is executed. This loop consists of:
  1) For each cycle t, each robot i (both leader and followers) updates its first potential $\mathbf{W}_i^t$ including the other robots in their positions $(x_j, y_j, z_j) \forall j \neq i$ as black points, representing obstacles. This step, is deeply explained in [16].
  2) According to the leader's pose and the desired formation geometry, the partial goal (x_{gf}, y_{gf}, z_{gf}) is calculated for each follower f (where f represents all the followers of the formation). Initially, the partial goals are computed with the predefined geometry. The gray level of these positions is used in order to recompute the partial goal as detailed in subsection III-C. Therefore, the shape of the formation is deformed so that the robots move farther from obstacles and the other robots, which are treated as obstacles. This way, the repulsive force between robots and walls and also the repulsive force between robots are implemented.
  3) The potentials $\mathbf{D}_i^t$ are calculated applying the FMM to velocities maps $\mathbf{W}_i^t$. For the leader the goal point is the end point of the path. In the case of the followers, they move towards the partial goals computed on the previous step. The low computational cost of FM^2 allows us to do this without compromising the refresh rate.
  4) The path is calculated for each robot i. This path

is the one with the minimum distance with the metrics $\mathbf{W}_i^t$ and it is obtained applying gradient descent on the potential $\mathbf{D}_i^t$.

5) All the robots move forward following their paths until a new iteration is completed.

C. Gray-level-based shape deformation

When the formation gets close to obstacles, the positions of the robots should be modified in order to adapt the predefined shape with the aim of avoiding collisions. The proposed method achieves this by modifying the distances of the predefined geometry. However, not all distances have to be changed in the same way for all the followers. Figure 7 shows the most simple deformation rules than can be used. The value *max* represents the distances of the original shape and *min* the minimum distances the formation will deform into. In this case, for the distances in directions parallel to the Frenet trihedron vectors $\mathbf{B}$ and $\mathbf{N}$, namely d_B and d_n, the behaviour is the same: the less gray level means the closer to obstacles, therefore this distances have to decrease towards the minimum (which is never reached since it means a collision). This means that the robots in the formation get closer to the leader's path, which is collision free (assuming static environments).

It could happen that an excessive contraction of the formation causes the followers to crash into each other. Therefore, priorities are introduced in the formation by modifying the distance in the direction parallel to vector $\mathbf{T}$, d_T. In this case, how much this distance varies depends on the robot. According to figure 7, follower 4 will not modify its distance, while follower 1 will be the one which gets closest to the leader. Therefore, highest priority is given by a higher slope in this function. Followers behaviour can be modified by setting different functions.

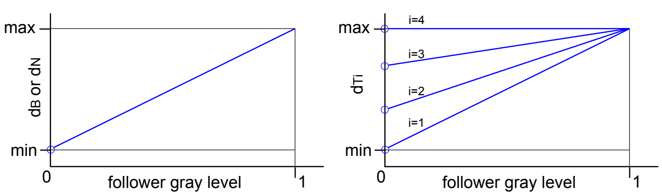

Fig. 7: Followers' partial goals modification based on linear functions.

IV. Results

According to the algorithm previously defined, different simulations have been carried out in order to prove the validity of the proposed method. All of them are based on Matlab® implementations of the algorithm and perform only kinematic simulation, meaning that we do not consider any specific dynamic model of the UAVs, and that a perfect path follow algorithm is assumed for all the UAVs. In all the cases, both the initial and the final points of the trajectory are given, and the paths are calculated with the FM^2 algorithm. To calculate the partial goals of the followers, a shape is previously set (e. g. a pyramid) defining the distances from the followers to the leader and modifying those distances as a function of the gray level of the current position.

Figure 8 shows an example of the algorithm performing in a very complex scenario: 3 narrow passages in opposed places of the map. In this figure, the leader is plotted as a green point, its path is the red line and the followers are cones representing their reference poses (for the sake of simplicity). The trajectories of the followers are represented with points. It is possible to appreciate how the formation *contracts* when going through the narrow passages and it is able to recover its shape in clear spaces. This is possible due to the variations of distances d_N and d_B proportionally to the gray level.

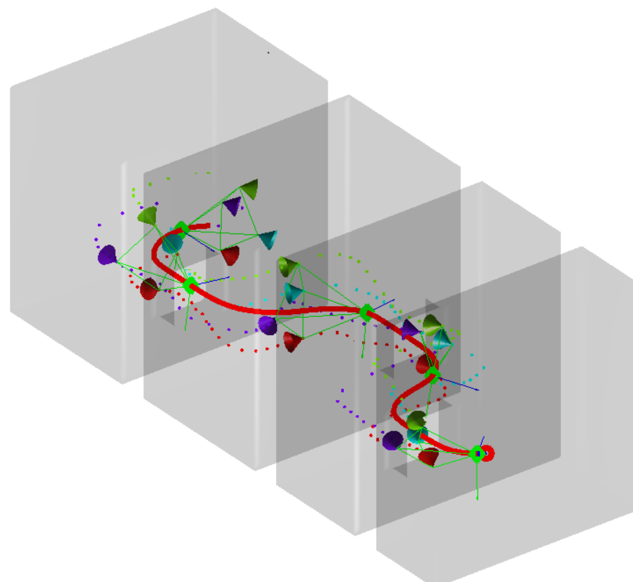

Fig. 8: Example of a motion sequence in a complex environment.

Figure 9 includes an example in which the priorities of the formation become extremely important. The narrow corridor forces the followers to be close to each other for a long time. Therefore, modifying also distance d_T it is possible to get extra space between followers.

V. Conclusion

This research introduces a novel approach to solve 3D robot formation motion planning problem. All the tests show that the proposed method, in combination with the FM^2 path planner, is robust enough to manage autonomous movements through an indoor static 3D environment.

It is important to note that the algorithm is both conceptual and mathematically very simple, since it relies on basic natural behaviours such as light movement.

Results show that the proposed algorithm is able to manage with difficult environments, modifying the formation when it is necessary. In addition, this approach allows us to include any number of robots in the formation, by only setting the desired position with respect to the leader or the other robots. The introduction of function-based geometry deformation is very powerful, since it allows to set very

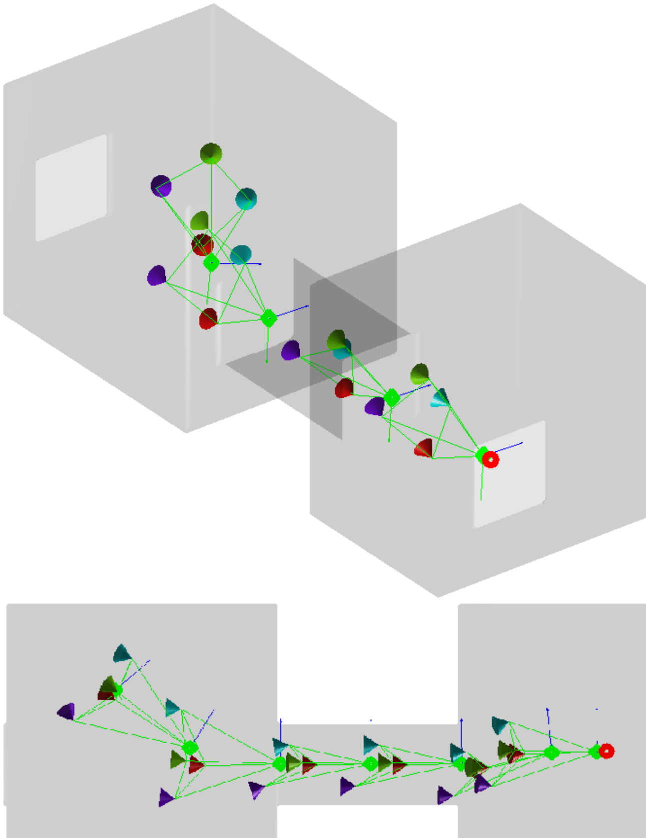

Fig. 9: Example of a motion sequence in a narrow corridor, leveraging the priorities introduced in the algorithm. Top: 3D view. Bottom: top-view which allows to see how the robots change their relative position in the direction of the motion.

complex behaviours to the followers by simply modifying the functions. As an example of this, the use of priorities in the formation is showed. These functions can be modified dynamically, an important property that is worthy to explore in the future.

Future work in 3D robot formation using FM^2 is also related to testing this method in dynamic environments in which uncertainties play an important role. Besides, other type of formations in which the leader is not always in front of the team are to be evaluated. Also, future simulation will include dynamics in order to prove that the computed paths are smooth enough to be applied to real robots.

ACKNOWLEDGEMENT

This work is funded by the project number DPI2010-17772, by the Spanish Ministry of Science and Innovation, and also by RoboCity2030-II-CM project (S2009/DPI-1559), funded by Programas de Actividades I+D en la Comunidad de Madrid and co-funded by Structural Funds of the EU.

REFERENCES

[1] S. K. M. Martin, P. Klupar, and J. Winter, TechSat 21 and Revolutionizing Space Missions Using Microsatellites, in *Proc. of USU/AIAA Conf.*, 2001.

[2] Dewan, A., Mahendran, A., Soni, N., Krishna, K.M., Heterogeneous UGV-MAV exploration using integer programming, *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 2013.

[3] S. Hauert, J.C. Zufferey and D. Floreano, Reverse-engineering of Artificially Evolved Controllers for Swarms of Robots, *IEEE Congress on Evolutionary Computation*, 2009.

[4] J.J. Acevedo, B.C. Arrue, I. Maza and A. Ollero, Cooperative Large Area Surveillance with a Team of Aerial Mobile Robots for Long Endurance Missions, *J. of Intelligent and Robotic Systems*, 2012.

[5] D. Thakur, M. Likhachev, J. Keller, V. Kumar, et al., Planning for Opportunistic Surveillance with Multiple Robots, *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 2013.

[6] S. Bouabdallah, Design and Control of Quadrotors with Applicationto Autonomous Flying, Theses 3727, EPFL, 2007.

[7] S. Hrabar, Reactive Obstacle Avoidance for Rotorcraft UAVs, *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pp. 4967–4974, 2011.

[8] S. Shen and N. Michael and V. Kumar, 3D Estimation and Control for Autonomous Flight with Constrained Computation, *IEEE Intl. Conf. of Robotics and Automation*, 2011.

[9] T. Hino, Simple Formation Control Scheme Tolerant to Communication Failures for Small Unmanned Air Vehicles, *Intl. Congress of the Aeronautical Sciences*, 2010.

[10] P. Ogren, M. Egerstedt, X. Hu, A control Lyapunov function approach to multiagent coordination, *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 847–851, 2002.

[11] M. Zhang, Y. Shen, Q. Wang, Y. Wang, Dynamic artificial potential field based multi-robot formation control, *IEEE Instrumentation and Measurement Technology Conf.*, 2010.

[12] Z. Cao, L. Xie, B. Zhang, S. Wang, M. Tan, Formation constrained multi-robot system in unknown environments, in *Proc. Int. Conf. on Robotics and Automation*, vol. 1, pp. 735–740, 2003.

[13] K.H. Tan, M.A. Lewis, Virtual Structures for High-Precision Cooperative Mobile Robotic Control, *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 1996.

[14] W. Ren and R.W. Beard, Decentralized Scheme for Spacecraft Formation Flying via the Virtual Structure Approach, *AIAA J. of Guidance, Control and Dynamics*, 2004.

[15] W. Yu, G. Chen, and M. Cao, Distributed leaderfollower flocking control for multi-agent dynamical systems with time-varying velocities, *Systems and Control Letters*, vol. 59, no. 9, pp. 543–552, 2010.

[16] J. V. Gómez, A. Lumbier, S. Garrido and L. Moreno, Planning Robot Formations with Fast Marching Square Including Uncertainty Conditions, *Robotics and Autonomous Systems*, pp. 137–152, vol. 61, no. 2, 2013.

[17] D. Álvarez, A. Lumbier, J.V. Gómez, S. Garrido and L. Moreno, Precision Grasp Planning with Gifu Hand III based on Fast Marching Square, *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 2013.

[18] A.Valero; J. V.Gómez; S.Garrido; L.Moreno. The Path to Efficiency. *IEEE Robotics and Automation Magz.*, vol. 20, no. 4, pp. 111-120. 2013.

[19] J. A. Sethian, A Fast Marching Level Set Method for Monotonically Advancing Fronts, *Proc. National Academy of Science*, pp. 1591–1595, vol. 93, no. 4, 1996.

[20] S. Osher and J. A. Sethian, Fronts Propagating with Curvature-Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations, *J. of Computational Physics*, no. 79, pp. 12–49, 1988.

[21] S. Garrido, L. Moreno, M. Abderrahim and D. Blanco, FM2: A Real-time Sensor-based Feedback Controller for Mobile Robots, *Intl. J. Robotics and Automation*, pp. 3169–3192, vol. 24, no. 1, 2009.

[22] L. Yatziv, A. Bartesaghi and G. Sapiro, A Fast O(n) Implementation of the Fast Marching Algorithm, *J. of Computational Physics*, pp. 393-399, vol. 212, no. 1, 2005.

[23] S. Garrido, L. Moreno, J.V. Gómez and P.U. Lima, General Path Planning Methodology for Leader-Followers based Robot Formations, *Intl. J. of Advanced Robotic Systems*, Vol. 10, No. 64, pp. 1 - 10, 2013.

[24] J.V. Gómez, S. Garrido and L. Moreno, Adaptive Robot Formations using Fast Marching Square working under Uncertainty Conditions, *IEEE Workshop on Advanced Robotics and Its Social Impacts*, pp. 68-71, California, USA. October, 2011.

[25] Frenet, F., Sur les Courbes à Double Courbure, *J. de Mathématiques Pures et Appliquées*, No. 17, pp. 437–447, vol. 1, 1852.

[26] Serret, J. A., Sur Quelques Formules Relatives à la Théorie des Courbes à Double Courbure, *J. de Mathématiques Pures et Appliquées*, no. 16, pp. 193–207, vol. 1, 1851.