2011 50th IEEE Conference on Decision and Control and
European Control Conference (CDC-ECC)
Orlando, FL, USA, December 12-15, 2011

Completely decentralised Navigation Functions for agents with finite sensing regions with application in aircraft conflict resolution

Giannis Roussos and Kostas J. Kyriakopoulos

Abstract— **We develop the framework of decentralised Navigation Functions for the case of multiple agents of arbitrary shapes and sensing areas around them moving in N-dimensional space. Unlike previous approaches utilising the Navigation Functions methodology, the construction here does not rely on diffeomorphisms , thus reducing the computational cost of the algorithm. The resulting potential field is absolutely locally computable and can be easily adapted to decentralised aircraft conflict resolution, as shown in the paper. Simulation results of multi-aircraft conflict resolution are included to support the efficacy of the complete control scheme.**

I. INTRODUCTION

The problem of navigation and motion planning for single, as well as for multiple agents is of great interest to a wide class of applications, ranging from the Control domain to Robotics and Air Traffic Management (ATM). Conflict avoidance and convergence to the target are the key requirements in such problems, while communication and performance constraints may apply.

Navigation Functions (NFs) have been proposed in [1] as an integrated solution to the path and motion planning problems, initially for the case of a single robot inside a bounded environment with stationary obstacles. The foundation of the NFs methodology emerged as an artificial potential field approach [2], where collision avoidance is achieved via repulsion between the robot and the obstacles, while an attractive component towards the destination is used for convergence. The key advantage of NFs is that unlike other potential field methods, no local minima appear in the potential field. Thus, a hill descent control law can be used without the risk of stagnation to undesired local minima. The first form of the methodology applied strictly to sphere worlds, i.e. problems where all obstacles, the workspace and the robot itself are perfect spheres. Subsequently, the authors in [3] presented a strategy for the construction of appropriate diffeomorphisms that transform star-shaped obstacles and *trees of stars* to spheres, so that the construction of [1] can then be applied. Because of the invariance of NFs's properties under diffeomorphisms, the performance guarantees are valid in this more general class of problems. An alternative approach exploiting the same property of NFs is presented in [4], where each obstacle and the robot is transformed to a point and the potential is built on the resulting point world. However, the construction of the appropriate diffeomorphisms can become computationally challenging, especially when the concept of *trees of stars* is employed to handle complex obstacles that cannot be represented by a single star-shape. Moreover, the deformation of the physical space introduces an additional logical step in the construction process and makes the tuning of the parameters involved more demanding.

NFs for multiple agents have been presented initially for a centralised control topology, assuming a sphere world, in [5]. For more general shapes, the algorithm relies on diffeomorphisms to appropriately transform the physical workspace into a sphere world. Extensions of this work to the decentralised case have also followed this approach, using sphere worlds for the construction of the potential fields [6]. In this approach, decentralisation is introduced by limiting the knowledge of each agent only to its own destination, thus the destinations of all other agents are unknown.

Limited sensing is a key factor for decentralisation: it allows the use of onboard sensors with finite range and greatly limits the information that each agent needs to acquire and process, significantly improving the applicability and scalability of the algorithm in large scenarios. Limited sensing so far has been introduced in a number of ways in NFs. In [7] the authors use a $\mathcal{C}^0$ sensing scheme, but assume a priori knowledge of the total number of agents. This requirement is removed in [8], where a switching sensing graph is used, resulting in a hybrid system. This approach does not ensure global convergence, as blocking situations may be reached. Thus, convergence occurs only if the switching of the sensing graph eventually stops. A completely locally computable NF has been presented in [9], but only for single-agent problems and with the assumption that at each time instant there is at most one visible obstacle. This effectively means that the algorithm solves the collision with one obstacle at a time, which is not practical in a multi-agent scenario. An initial approach towards the application of local sensing in a continuous manner for multiple obstacles has been presented in [10] for a circular sensing area, while a non-circular sensing scheme has been introduced in [11] that improves the quality of the resulting paths.

This work presents a generalisation of the NF construction in scenarios with non-circular agents without the need for diffeomorphisms, while local sensing is also implemented via non-circular sensing zones. The construction of the potential relies on $\mathcal{C}^2$ functions that implicitly define the boundaries of the agents and their sensing areas in the N-dimensional space. This general and flexible form in which agent and sensing area shapes are described allows the tailoring of the methodology to the specific characteristics of aircraft conflict

Giannis Roussos and Kostas J. Kyriakopoulos are with the Control Systems Lab, Department of Mechanical Engineering, National Technical University of Athens, 9 Heroon Polytechniou Street, Zografou 15780, Greece. email: {jrous, kkyria}@mail.ntua.gr

resolution. The significant difference between horizontal and vertical separation minima in Air Traffic Control (ATC) can be easily taken into account by the algorithm. Moreover, a specially chosen non-circular sensing area can offer improved trajectories, as already demonstrated for the 2D case in [11].

The rest of the paper is organised as follows: in section II we review the basic structure of decentralised Navigation Functions (NFs) that we employ, along with its components. In the next section the proposed construction for the collision function is presented for the general case of non-circular shapes. In section IV the adaptation of the algorithm to the problem of aircraft conflict resolution in 3D space is developed. Simulation results are presented in section V, followed by our conclusions in section VI.

II. Completely Decentralised Navigation Functions

The NF construction we present here is intended to cover a general class of problems where a group of N agents of various shapes and sensing characteristics operate in a common environment. Specifically, our work here focuses on the construction of the collision function β_{ij}, which is a key component of the complete potential. The proposed approach inherently takes into account the shape of each agent i, O_i, and the shape of the sensing region A_i, within which it can detect other agents and obstacles, as in Fig. 1.

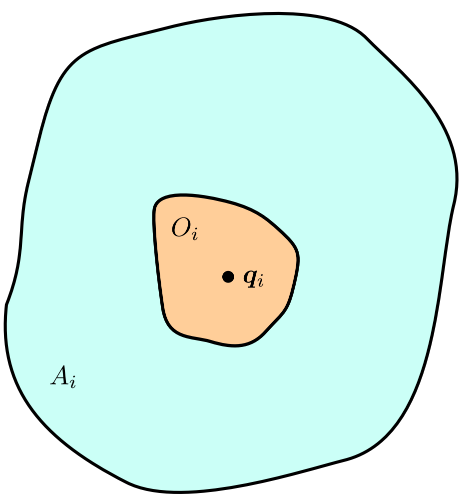


Fig. 1: Agent i located at $\boldsymbol{q}_i$. Its shape is represented by O_i and its sensing area by A_i

The basic form of the decentralised NF Φ_i for agent i is based on previous multi-agent NF approaches [6]:

$$\Phi_i = \frac{\gamma_{di} + f_i}{\left((\gamma_{di} + f_i)^k + G_i \cdot \beta_{0i}\right)^{1/k}}, \tag{1}$$

where G_i quantifies the proximity to collisions involving agent i: it is zero when the i^{th} agent participates in a conflict, and positive otherwise. γ_{di} is the goal function, fading at the destination of agent i, $\boldsymbol{q}_{id}$, and increasing away from it. Function $f_i = f_i(G_i)$ enables cooperation between neighboring agents as explained in [6], while β_{0i} is the workspace bounding obstacle that limits the motion of agents inside the available workspace.

Our contribution in this paper, as described in the next section, lies mostly in the construction of the individual collision functions β_{ij} that model the effect of other agents and obstacles to agent i. The product of all β_{ij} forms the overall collision function G_i that is used in the potential:

$$G_i = \prod_{j \in T_i} \beta_{ij} \tag{2}$$

where $T_i = \{1, \ldots, N\} \setminus i$.

For the target function γ_i we use the normalised form introduced in [10]:

$$\gamma_i = \frac{\|\boldsymbol{q}_i - \boldsymbol{q}_{di}\|^2}{R_n^2} \tag{3}$$

where R_n is a reference length used for the normalisation that reflects the physical scale of the problem, eg. the largest dimension of the workspace.

The cooperation function f_i is used here as in [7]:

$$f_i(G_i) = \begin{cases} a_0 + \sum_{l=1}^{3} a_l G_i^l, & G_i \leq X \\ 0, & G_i > X \end{cases} \tag{4}$$

where $a_0 = Y$, $a_1 = 0$, $a_2 = \frac{-3Y}{X^2}$, $a_3 = \frac{2Y}{X^3}$ and X, Y are positive parameters. Values of G_i lower than X activate the cooperation function f_i, while Y defines the maximum value of f_i, which is attained when $G_i = 0$.

III. Constructing the collision function

The repulsive function β_{ij} introduced here is redesigned to handle a more general class of shapes for the obstacles as well as the agents and their sensing regions. Instead of building the NF potential on a virtual sphere world, we adapt the methodology to real shapes that can be implicitly defined through the level sets of some functions.

A. Implicit obstacle description

In order to derive the contribution of an intruding agent j to agent i's potential Φ_i, β_{ij} , we use an *implicit obstacle function* δ_{ij} as in [12], such that the boundary of collision between agents i and j corresponds to its zero level set:

$$O_{ij} \triangleq \{(\boldsymbol{q}_i, \boldsymbol{q}_j) | \delta_{ij} \leq 0\}$$
$$\partial O_{ij} = \{(\boldsymbol{q}_i, \boldsymbol{q}_j) | \delta_{ij} = 0\}$$

where O_{ij} is the set that represents all possible collisions between agents i and j. The obstacle function δ_{ij} is a measure of collision between agents i and j: it is negative when the agents are in collision, fades to 0 when they touch and becomes positive when they are separated.

In applications where the collision avoidance requirements are expressed directly in terms of the relative position $\boldsymbol{q}_{ij} = \boldsymbol{q}_j - \boldsymbol{q}_i$ between agents, the collision set O_{ij} can be described via a *relative obstacle function* $\tilde{\delta}_{ij}(\boldsymbol{q}_{ij})$, such that $\tilde{\delta}_{ij}(\boldsymbol{q}_{ij}) \leq 0$ when agents i and j collide, i.e. $(0, \boldsymbol{q}_{ij}) \in O_{ij}$. The implicit obstacle function $\delta_{ij}(\boldsymbol{q}_j)$ of the absolute position $\boldsymbol{q}_j$ can be then derived directly as:

$$\delta_{ij}(\boldsymbol{q}_i, \boldsymbol{q}_j) = \tilde{\delta}_{ij}(\boldsymbol{q}_j - \boldsymbol{q}_i)$$

Alternatively, the shape $\tilde{O}_i$ of each individual agent i may be given as the zero level set of a relative shape function $\tilde{\delta}_i$:

$$\tilde{O}_i \triangleq \left\{\boldsymbol{q}^i | \tilde{\delta}_i(\boldsymbol{q}^i) \leq 0\right\}$$
$$\partial\tilde{O}_i = \left\{\boldsymbol{q}^i | \tilde{\delta}_i(\boldsymbol{q}^i) = 0\right\}$$

where $\boldsymbol{q}^i = \boldsymbol{q} - \boldsymbol{q}_i$ is the position in the local frame of agent i, originated at $\boldsymbol{q}_i$. For an arbitrary position of agent i the absolute shape function $\delta_i(\boldsymbol{q}_i, \boldsymbol{q})$ defines the shape O_i of agent i in the global frame, via a translation of $\tilde{O}_i$ by $\boldsymbol{q}_i$:

$$\delta_i(\boldsymbol{q}_i, \boldsymbol{q}) = \tilde{\delta}_i(\boldsymbol{q} - \boldsymbol{q}_i)$$
$$O_i(\boldsymbol{q}_i) \triangleq \{\boldsymbol{q} : \delta_i(\boldsymbol{q}_i, \boldsymbol{q}) \leq 0\}$$
$$\partial O_i(\boldsymbol{q}_i) = \{\boldsymbol{q}^i : \delta_i(\boldsymbol{q}_i, \boldsymbol{q}) = 0\}$$

The collision set O_{ij} between two agents i and j will comprise all $(\boldsymbol{q}_i, \boldsymbol{q}_j)$ pairs that cause $O_i(\boldsymbol{q}_i)$ and $O_j(\boldsymbol{q}_j)$ to overlap. The intersection of $O_i(\boldsymbol{q}_i)$ and $O_j(\boldsymbol{q}_j)$ can be implicitly represented using the formula given in [13]:

$$\psi_\cap(\delta_i, \delta_j) \triangleq \delta_i + \delta_j + \sqrt{\delta_i^2 + \delta_j^2}$$
$$O_i(\boldsymbol{q}_i) \cap O_j(\boldsymbol{q}_j) = \{\boldsymbol{q} | \psi_\cap(\delta_i(\boldsymbol{q}_i, \boldsymbol{q}), \delta_j(\boldsymbol{q}_j, \boldsymbol{q})) \leq 0\}$$
$$\partial(O_i(\boldsymbol{q}_i) \cap O_j(\boldsymbol{q}_j)) = \{\boldsymbol{q} | \psi_\cap(\delta_i(\boldsymbol{q}_i, \boldsymbol{q}), \delta_j(\boldsymbol{q}_j, \boldsymbol{q})) = 0\}$$

Therefore, the obstacle set O_{ij} will then comprise all $(\boldsymbol{q}_i, \boldsymbol{q}_j)$ pairs that result in an non empty $O_i(\boldsymbol{q}_i) \cup O_j(\boldsymbol{q}_j)$, i.e. $(\boldsymbol{q}_i, \boldsymbol{q}_j) \in O_{ij} \iff O_i \cap O_j \neq \emptyset$:

$$O_{ij} = \left\{(\boldsymbol{q}_i, \boldsymbol{q}_j) | \min_{\boldsymbol{q}} \psi_\cap(\delta_i(\boldsymbol{q}_i, \boldsymbol{q}), \delta_j(\boldsymbol{q}_j, \boldsymbol{q})) \leq 0\right\}$$

The boundary ∂O_{ij} will comprise all $(\boldsymbol{q}_i, \boldsymbol{q}_j)$ pairs that cause $O_i(\boldsymbol{q}_i)$ and $O_j(\boldsymbol{q}_j)$ to touch, i.e. all their common points will be on $\partial O_i(\boldsymbol{q}_i) \cup \partial O_j(\boldsymbol{q}_j)$:

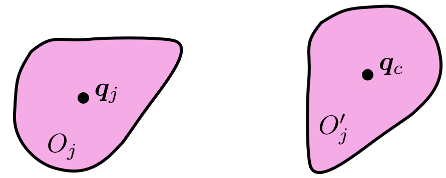
$$\partial O_{ij} = \left\{(\boldsymbol{q}_i, \boldsymbol{q}_j) | \min_{\boldsymbol{q}} \psi_\cap(\delta_i(\boldsymbol{q}_i, \boldsymbol{q}), \delta_j(\boldsymbol{q}_j, \boldsymbol{q})) = 0\right\}$$

For all $(\boldsymbol{q}_i, \boldsymbol{q}_j)$ pairs such that $O_i(\boldsymbol{q}_i)$ and $O_j(\boldsymbol{q}_j)$ overlap there is at least one common point $\boldsymbol{q}_c$ that belongs to both of them, thus $\delta_i(\boldsymbol{q}_i, \boldsymbol{q}_c) \leq 0$ and $\delta_j(\boldsymbol{q}_j, \boldsymbol{q}_c) \leq 0$. Rewriting this with $\boldsymbol{q}_c$ as a reference point and using $\boldsymbol{q}_j^c = \boldsymbol{q}_j - \boldsymbol{q}_c$ as the relative position vector of $\boldsymbol{q}_j$ with respect to $\boldsymbol{q}_c$ we obtain:

$$\delta_j(\boldsymbol{q}_j, \boldsymbol{q}_c) = \tilde{\delta}_j(-(\boldsymbol{q}_j - \boldsymbol{q}_c)) = \tilde{\delta}(-\boldsymbol{q}_j^c) \leq 0 \tag{5}$$

which means that $\boldsymbol{q}_j$ belongs to the set $O'_j(\boldsymbol{q}_c)$, produced by mirroring $\tilde{O}_j$ with respect to the hyperplane that is normal to the N-dimensional vector of ones $\vec{\mathbf{1}} = \begin{bmatrix} 1 & 1 & \dots & 1 \end{bmatrix}$, and then translating it to $\boldsymbol{q}_c$, as shown in Figure 2. Thus, the set of all $\boldsymbol{q}_j$ such that O_i, O_i have at least one common point $\boldsymbol{q}_c$ can be composed of all $O'_j(\boldsymbol{q}_c)$ for every $\boldsymbol{q}_c \in O_i$:

$$O_{ij} = \bigcup_{\boldsymbol{q}_c \in O_i} O'_j(\boldsymbol{q}_c) \tag{6}$$



(a) (b)

Fig. 2: (a): The shape O_j of agent j around its position $\boldsymbol{q}_j$
(b): The shape $O'_j(\boldsymbol{q}_c)$ derived by mirroring O_j around the $y = -x$ line and translating it to $\boldsymbol{q}_c$

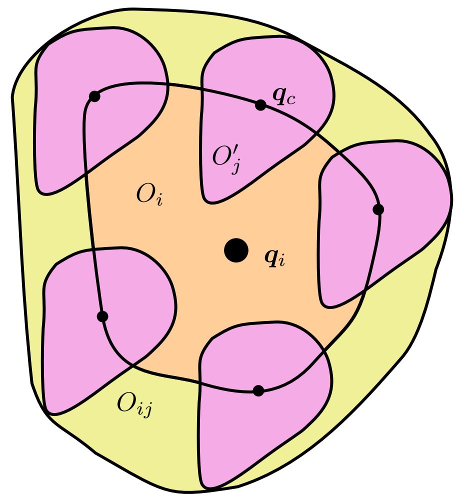


Fig. 3: Construction of the collision set O_{ij} between agents i, j, by sliding the set $O'_j(\boldsymbol{q}_c)$ along the boundary of O_i. (O_i has been assumed larger than O_j for the sake of a cleaner figure.)

This allows us to construct O_{ij} graphically, as shown in Figure 3, by taking the mirrored shape O'_j around $\boldsymbol{q}_c$ and deriving the Minkowski sum of O_i and O_j, by sliding $\boldsymbol{q}_c$ along the boundary of O_i.

Identifying an obstacle function δ_{ij} such that $\delta_{ij}(\boldsymbol{q}_i, \boldsymbol{q}_j) \leq 0 \Leftrightarrow (\boldsymbol{q}_i, \boldsymbol{q}_j) \in O_{ij}$ and $\delta_{ij}(\boldsymbol{q}_i, \boldsymbol{q}_j) = 0 \Leftrightarrow (\boldsymbol{q}_i, \boldsymbol{q}_j) \in \partial O_{ij}$ is not straightforward in general. Following the previous analysis, one may suggest the use of $\min \psi_\cap(\delta_i, \delta_j)$. Although this option is valid, it is not always practical, since the minimum of $\psi_\cap(\delta_i, \delta_j)$ is not easy to derive in general, especially through analytic calculations. However, this does not prevent the use of the method in ATM, since conflict avoidance constraints in this application are expressed with respect to the relative position of neighboring aircraft. Thus, δ_{ij} can be directly derived and the methodology described in the previous paragraph is immediately applicable.

B. Local Sensing

A limited sensing region $\tilde{A}_i$ around agent i is used to restrict the sensing of other agents and obstacles by agent i: only those agents and obstacles that are inside $\tilde{A}_i$ can influence the potential Φ_i. Similarly to $\tilde{O}_i$ above, $\tilde{A}_i$ is defined in the local coordinate frame of agent i via the zero level set of the relative sensing function $\tilde{s}_i(\boldsymbol{q}^i)$:

$$\tilde{A}_i \triangleq \{\boldsymbol{q}^i : \tilde{s}_i(\boldsymbol{q}^i) = 0\} \tag{7}$$
$$\partial\tilde{A}_i = \{\boldsymbol{q}^i : \tilde{s}_i(\boldsymbol{q}^i) = 0\} \tag{8}$$

In global coordinates the absolute sensing region A_i is described by the *implicit sensing function* $s_i(\boldsymbol{q}_i, \boldsymbol{q}) = \tilde{s}_i(\boldsymbol{q} - \boldsymbol{q}_i)$:

$$A_i(\boldsymbol{q}_i) \triangleq \{\boldsymbol{q} : s_i(\boldsymbol{q}_i, \boldsymbol{q}) = 0\} \tag{9}$$

$$\partial A_i(\boldsymbol{q}_i) = \{\boldsymbol{q} : s_i(\boldsymbol{q}_i, \boldsymbol{q}) = 0\} \tag{10}$$

Obviously, each agent i should be able to sense another agent or obstacle j before an actual collision between them occurs. Thus, the set $O^i_j(\boldsymbol{q}_i) = \{\boldsymbol{q}| (\boldsymbol{q}_i, \boldsymbol{q}) \in O_{ij}\}$, which is the set of all $\boldsymbol{q}_j$ that cause a collision for a given position $\boldsymbol{q}_i$ of agent i, must always be completely contained inside the sensing region A_i for all pairs of i, j. When inside the set $A_i(\boldsymbol{q}_i) \setminus O^i_j(\boldsymbol{q}_i)$, the agent (or obstacle) j can influence the potential Φ_i. Thus the properties of the functions δ_i and s_i, which will be used to derive β_{ij}, inside this region are important for the overall behavior of the potential Φ_i.

C. Collision function synthesis

As explained above, the collision function β_{ij} is active only when $\boldsymbol{q}_j \in A_i(\boldsymbol{q}_i) \setminus O^i_j(\boldsymbol{q}_i)$, i.e. inside A_i where $s_i \leq 0$, and outside O^i_j, where $\delta_{ij} \geq 0$. Consequently, we can use the "conditioning" diffeomorphism σ_λ from [12] to map $-\frac{\delta_{ij}}{s_i} \in [0, +\inf]$ to $[0, 1]$:

$$\bar{\beta}_{ij} \triangleq \left(\sigma_\lambda \circ -\frac{\delta_{ij}}{s_i}\right) = \frac{\delta_{ij}}{\delta_{ij} - \lambda s_i} \tag{11}$$

$$\text{where } \sigma_\lambda \triangleq \frac{x}{\lambda + x}, \quad \lambda > 0$$

By the above definition, $\bar{\beta}_{ij}$ is zero when agents i, j touch (i.e. when $\delta_{ij} = 0$ because $(\boldsymbol{q}_i, \boldsymbol{q}_j) \in \partial O_{ij}$), and increases up to 1 at the boundary of the sensing area A_i where $s_i = 0$.

When $\boldsymbol{q}_j$ is outside the sensing region A_i, the influence β_{ij} should be inactive by assuming a constant value of 1. Thus, in order to ensure that β_{ij} will be $\mathcal{C}^2$, we apply a shaping function $L(x)$ to $\bar{\beta}_{ij}$:

$$L(x) = x^3 - 3x^2 + 3x \tag{12}$$

The following properties hold for $L(x)$:

$$\begin{aligned} L(0) &= 0 & \qquad L'(x) &> 0 \ \forall x \in [0, 1) \\ L(1) &= 1 & \qquad L'(1) &= L''(1) = 0 \end{aligned}$$

Thus, the final collision function used is:

$$\beta_{ij} = \begin{cases} L\left(\bar{\beta}_{ij}\right), & \bar{\beta}_{ij} \leq 1 \\ 1, & \bar{\beta}_{ij} > 1 \end{cases} \tag{14}$$

D. The workspace bounding obstacle

The function β_{0i} models a special obstacle which ensures that all agents remain inside the available workspace $W \subset \mathcal{R}^n$. For its construction we follow an similar approach to the one presented for β_{ij} above. We assume that the workspace is implicitly defined through a scalar function $\delta_0(\boldsymbol{q})$:

$$W \triangleq \{\boldsymbol{q}|\delta_0(\boldsymbol{q}) \leq 0\} \tag{15}$$

$$\partial W \triangleq \{\boldsymbol{q}|\delta_0(\boldsymbol{q}) = 0\} \tag{16}$$

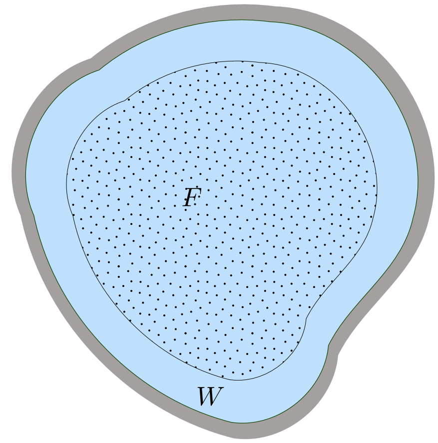


Fig. 4: The workspace W (blue color) and the region F (dotted pattern) where its influence is zero. Thus, agents take into account the workspace boundary only within the blue undotted annulus.

Moreover, we define a region F contained inside W, where the influence of the workspace boundary is eliminated:

$$F \triangleq \{\boldsymbol{q}|s_0(\boldsymbol{q}) \leq 0\} \tag{17}$$

$$\partial F = \{\boldsymbol{q}|s_0(\boldsymbol{q}) = 0\} \tag{18}$$

Thus, we allow the agents to be affected by the workspace boundary only inside the annulus $W \setminus F$, as shown in Figure 4. In this region $\delta_{i0} = \delta_0(\boldsymbol{q}_i) \leq 0$ and $s_{i0} = s_0(\boldsymbol{q}_i) \geq 0$, thus, similarly to the synthesis of $\bar{\beta}_{ij}$, we use σ_λ to map $-\frac{\delta_{i0}}{s_{i0}} \in [0, +\inf)$ to $\bar{\beta}_{i0} \in [0, 1]$:

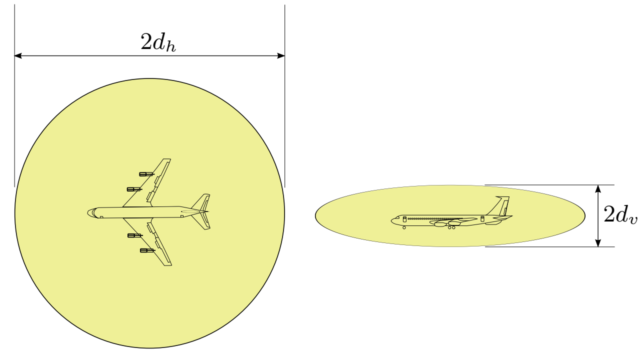
$$\bar{\beta}_{i0} \triangleq \left(\sigma_\lambda \circ -\frac{\delta_{i0}}{s_{i0}}\right) = \frac{\delta_{i0}}{\delta_{i0} - \lambda s_{i0}} \tag{19}$$

One can see that $\bar{\beta}_{i0}$ is zero on ∂W and becomes 1 on the boundary of the influence area, ∂F_0. Consequently, the workspace obstacle function is derived by applying the $L(x)$ mapping to $\bar{\beta}_{i0}$ to ensure it is $\mathcal{C}^2$:

$$\beta_{i0} = \begin{cases} L\left(\bar{\beta}_{i0}\right), & \bar{\beta}_{i0} \leq 1 \\ 1, & \bar{\beta}_{i0} > 1 \end{cases} \tag{20}$$

IV. Decentralised aircraft Conflict resolution in 3D space

The application considered in this paper is the decentralised navigation of a group of N independent aircraft in 3D airspace. Each aircraft is modeled as a kinematic agent with its configuration comprising its position $\boldsymbol{q}_i = \begin{bmatrix} x_i & y_i & z_i \end{bmatrix}^\top$ with respect to an earth-fixed frame $\mathcal{E}$, and its heading ϕ_i. The global x and y axes lie on the horizontal plane, while axis z is the vertical (altitude) axis. The heading angle ϕ_i is the angle of the aircraft's horizontal velocity with respect to the global x axis. Using $\boldsymbol{n}_i = \begin{bmatrix} x_i & y_i \end{bmatrix}^\top$ as the projection of the agent's position on the horizontal $x - y$



(a) (b)

Fig. 5: The collision set around each aircraft, defined as an oblate spheroid of horizontal radius d_h and vertical semi-axis d_v. d_h and d_v are the horizontal and vertical separation minima, respectively. (not drawn in scale)

plane, the kinematic model of each aircraft i is:

$$\begin{aligned} \dot{\boldsymbol{n}}_i = \begin{bmatrix} \dot{x}_i \\ \dot{y}_i \end{bmatrix} &= \mathbf{J}_i \cdot u_i, \\ \dot{z}_i &= w_i, \\ \dot{\phi}_i &= \omega_i, \end{aligned} \tag{21}$$

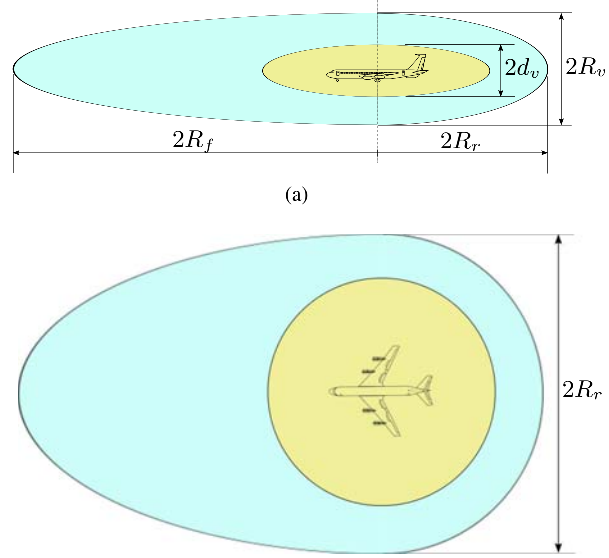
where $\boldsymbol{J}_i = \begin{bmatrix} \cos(\phi_i) & \sin(\phi_i) \end{bmatrix}^\top$.

A. Collision and sensing regions

The potential construction described in section II can be well adapted to the case of aircraft conflict resolution. Decentralisation, as well as the ability to use 3D resolutions to increase airspace capacity and separation are very useful for ATM applications, like the future ATM concept developed within the project iFLY [14].

In order to apply the methodology described in the previous sections to aircraft conflict resolution, we define accordingly the shapes used for the collision set O_{ij} and the sensing area A_i of each aircraft. For O_{ij} we employ a "pill-like" 3D oblate spheroid with horizontal radius equal to the minimum horizontal separation d_h and a vertical semi-axis equal to the vertical minimum separation d_v, as shown in Figure 5. The principle in [11] is extended to the 3D case for A_i: the sensing range in the forward direction is significantly longer than the sides and rear, as shown in Figure 6. The benefits of such a sensing scheme have been demonstrated in [11], allowing smoother turns and smaller deviations in the resulting trajectories. The sensing region comprises two half-ellipsoids, one in front of the aircraft with semi-axes R_f, R_r and R_v in the forward, side and vertical directions respectively, and one in the rear with semi-axes R_r, R_r and R_v. Thus, the two cross-sections on the plane normal to the major axis x of the aircraft match exactly, while a longer range R_w is applied forward compared to the shorter side and rear range R_r. As shown in [11], such a sensing scheme introduces *prioritisation*, since the interaction between two neighboring agents is unsymmetrical.

In order to model the shape of O_{ij} with the implicit



(a)

(b)

Fig. 6: The sensing region A_i around each aircraft, extending to R_f forward, R_r to the rear and sides and R_v on the vertical direction (not drawn in scale)

collision function δ_{ij} we use the standard ellipsoid formula:

$$\delta_{ij} = \frac{\left(x_j^i\right)^2}{d_h^2} + \frac{\left(y_j^i\right)^2}{d_h^2} + \frac{\left(z_j^i\right)^2}{d_v^2} - 1 \tag{22}$$

where $\boldsymbol{q}_j^i = \begin{bmatrix} x_j^i & y_j^i & z_j^i \end{bmatrix}$ is the position of aircraft j in the local coordinate frame of aircraft i; x_j^i is along the longitudinal direction of aircraft i, y_j^i along the lateral direction and z_j^i on the vertical one:

$$\boldsymbol{q}_j^i = T_i \left(\boldsymbol{q}_j - \boldsymbol{q}_i\right) \tag{23}$$

$$T_i = \begin{bmatrix} \cos(\phi_i) & \sin(\phi_i) & 0 \\ -\sin(\phi_i) & \cos(\phi_i) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{24}$$

Similarly, for the sensing function s_{ij} we use the form:

$$s_{ij} = \frac{\left(x_j^i\right)^2}{R_x^2} + \frac{\left(y_j^i\right)^2}{R_r^2} + \frac{\left(z_j^i\right)^2}{R_v^2} - 1 \tag{25}$$

$$R_x = \begin{cases} R_f, & x_j^i \geq 0 \quad \text{(agent } j \text{ in front of } i\text{)} \\ R_r, & x_j^i < 0 \quad \text{(agent } j \text{ behind } i\text{)} \end{cases} \tag{26}$$

B. 3D motion control scheme for aircraft-like vehicles

By applying the forms for s_{ij} and δ_{ij} to the NF construction described in II, we can create a potential that corresponds to the specific requirements of aircraft conflict resolution. In order to guide each aircraft i using the potential Φ_i, the control scheme presented by the authors in [15] can be employed. This control policy is specially designed to allow each aircraft to move towards lower potential values with a constant speed, while respecting climb and descent angle limits and reducing unnecessary steering.

V. Results

The control scheme presented here has been employed in a simulation scenario, corresponding to aircraft Short-term conflict resolution, according to ATM standards. The

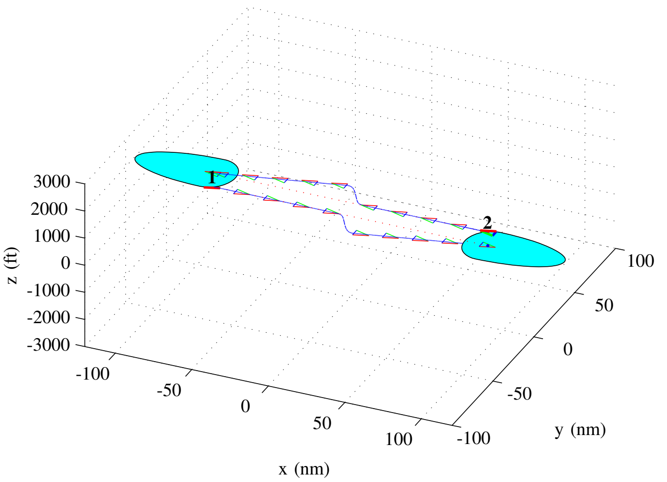


Fig. 7: Two aircraft with insufficient vertical separation perform vertical manoeuvres to resolve the conflict.

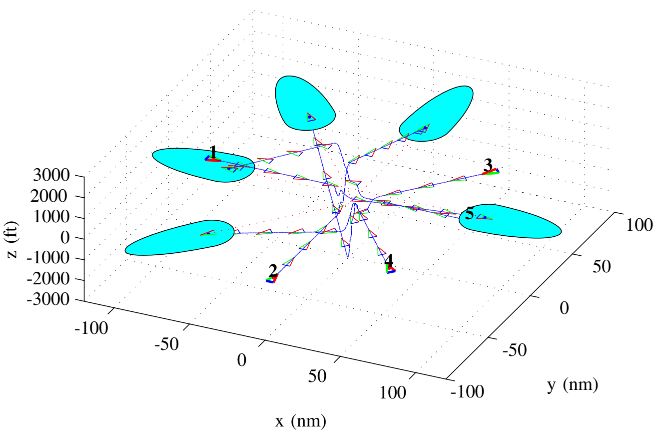


Fig. 8: Multiple aircraft 3D conflict resolution

horizontal separation minimum d_h has been set to 5 nautical miles (nm), while the vertical separation minimum used is 1000 feet (ft). The sensing ranges have been set as follows:

- $R_f = 40nm$ for the forward sensing range, corresponding to about 5 minutes of flight with a typical cruising speed of 480 knots. This is in accordance with the iFLY Concept of Operations [14] that specifies 5 minutes as the limit for Short-term conflict resolution.
- $R_r = 15nm$, reducing influence from other aircraft in the rear and sides to improve the trajectories.
- $R_v = 1500ft$, in order to avoid excessive separation by more than two flight levels ($2000ft$).

First, a simple scenario with 2 aircraft presents the principles of operation of the algorithm. The aircraft are flying in opposite direction with an initial vertical separation of only $500ft$. As shown in Figure 7, the aircraft perform vertical manoeuvres to increase their separation and then revert back to their desired altitude as they approach their goals.

The second scenario includes 5 aircraft flying at the same altitude in converging paths. As shown in Figure 8, vertical maneuvering enables safe resolution of all conflicts.

VI. Conclusions

We have presented a way to extend the application of Navigation Functions (NFs) to problems with agents of general shapes and sensing schemes. The scalar functions that implicitly describe the shapes of the sensing regions and the agents themselves are used to synthesise the obstacle functions for the potential. This NF construction enables the adaptation of the methodology to various applications by using appropriate shape functions. Application to the case of distributed aircraft conflict resolution is presented, for which NFs are a candidate solution. The specific requirements of this problem are considered for the definition of the agent and sensing region shapes. The algorithm is used in simulation scenarios to demonstrate its efficacy.

VII. Acknowledgements

The authors of this paper want to acknowledge the contribution of the European Commission through project iFLY.

References

[1] D. Koditschek and E. Rimon, “Robot navigation functions on manifolds with boundary,” *Advances in Applied Mathematics*, vol. 11, no. 4, pp. 412–442, 1990.
[2] O. Khatib, “Real-Time Obstacle Avoidance for Manipulators and Mobile Robots,” *The International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986.
[3] E. Rimon and D. E. Koditschek, “Exact robot navigation uing artificial potential functions,” *IEEE Transactions on Robotics and Automation*, vol. 8, no. 5, pp. 501–518, 1992.
[4] H. Tanner, S. Loizou, and K. Kyriakopoulos, “Nonholonomic navigation and control of cooperating mobile manipulators,” *Robotics and Automation, IEEE Transactions on*, vol. 19, pp. 53 – 64, Feb. 2003.
[5] S. Loizou and K. Kyriakopoulos, “Closed loop navigation for multiple holonomic vehicles,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, pp. 2861 – 2866 vol.3, 2002.
[6] D. V. Dimarogonas, S. G. Loizou, K. J. Kyriakopoulos, and M. M. Zavlanos, “A feedback stabilization and collision avoidance scheme for multiple independent non-point agents,,” *Automatica*, vol. 42, no. 2, pp. 229 – 243, 2006.
[7] D. V. Dimarogonas and K. J. Kyriakopoulos, “Decentralized navigation functions for multiple robotic agents with limited sensing capabilities,” *Journal of Intelligent and Robotic Systems*, vol. 48, no. 3, pp. 411–433, 2007.
[8] D. V. Dimarogonas, K. J. Kyriakopoulos, and D. Theodorakatos, “Totally distributed motion control of sphere world multi-agent systems using decentralized navigation functions,” *2006 IEEE International Conference on Robotics and Automation*, pp. 2430–2435, 2006.
[9] G. Lionis, X. Papageorgiou, and K. J. Kyriakopoulos, “Locally computable navigation functions for sphere worlds,” in *IEEE International Conference on Robotics and Automation*, pp. 1998–2003, 2007.
[10] G. Roussos and K. J. Kyriakopoulos, “Completely decentralised navigation of multiple unicycle agents with prioritization and fault tolerance,” in *49th IEEE Conference on Decision and Control*, 2010.
[11] G. Roussos and K. J. Kyriakopoulos, “Decentralized & prioritized navigation and collision avoidance for multiple mobile robots,” in *10th International Symposium on Distributed Autonomous Robotics Systems, Lausanne, Switzerland.*, 2010.
[12] E. Rimon and D. E. Koditschek, “Exact robot navigation using artificial potential functions,” *IEEE Transactions on Robotics and Automation*, vol. 8, no. 5, pp. 501–508, 1992.
[13] O. V. Zenkin, “Analytical description of geometrical shapes,” *Cybernetics and Systems Analysis*, vol. 6, pp. 481–489, 1970. 10.1007/BF01073248.
[14] G. Cuevas, I. Echegoyen, J. Garca, P. Csek, C. Keinrath, R. Weber, P. Gotthard, F. Bussink, and A. Luuk, “Autonomous Aircraft Advanced (A^3) Concept of Operations,” *iFLY Deliverable D1.3*, 2010.
[15] G. Roussos and K. J. Kyriakopoulos, “Decentralised navigation and collision avoidance for aircraft in 3D space,” *2010 American Control Conference, Baltimore, USA*, 2010.