ꟼ periodica polytechnica

Electrical Engineering and Computer Science
56/4 (2012) 113–124
doi: 10.3311/PPee.7080
http://periodicapolytechnica.org/ee
Creative Commons Attribution ⓘ

RESEARCH ARTICLE

Formation Control of Quadrotor Helicopters with Guaranteed Collision Avoidance via Safe Path

Gergely Regula / *Béla* Lantos

Received 2012-10-12, accepted 2013-02-25

Abstract

In this article we propose a hierarchical control structure for multi-agent systems. The main objective is to perform formation change manoeuvres, with guaranteed safe distance between each two vehicles throughout the whole mission. The key components that ensure safety are a robust control algorithm that is capable of stabilising the group of vehicles in a desired formation and a higher level path generation method that provides safe paths for all the vehicles, based on graph theoretic considerations. The method can efficiently handle a large group of any type of vehicles. In the article we focus on the control of quadrotor UAVs, thus the results are illustrated in 4D on a group of such vehicles.

Keywords

multi-agent system · formation control · UAV · quadrotor helicopters · robust control · distributed control

Acknowledgement

This research was supported by the Hungarian National Research Program under grant No. OTKA K71762 and by the MTA – BME Control Engineering Research Group.

The work of B. Lantos was supported by the European Union and Hungary in the project TÁMOP-4.2.2.A-11/1/KONV-2012-0012: „Basic research for the development of hybrid and electric vehicles”.

Gergely Regula
MTA – BME Control Engineering Research Group, Kende u. 13-17., H-1111 Budapest, Hungary
e-mail: regula@iit.bme.hu

Béla Lantos
Research Centre of Vehicle Industry, Széchenyi István University, Egyetem tér 1., H-9026 Győr, Hungary
e-mail: lantos@iit.bme.hu

1 Introduction

Increasing attention has been focused on the problem of controlling large scale systems that are built up from several smaller subsystems, e.g. a group of UAVs. Controlling a group of vehicles together can result in better overall performance and certain tasks can also be performed more effectively. Examples to such cases are surveillance missions, fuel consumption reduction by travelling in formation.

Advances in communication technology, miniaturisation and increased computation power open the way to implement not only local, but also formation level control algorithms on board of a single vehicle. Performing all the required calculations in a centralised manner is often not viable. In such cases, distributed solutions are required, even though additional problems arise, e.g. communication errors or delays.

Several methods have been elaborated that solve certain problems related to multi-vehicle systems. Each of them have strengths and weaknesses, thus they have evolved in parallel. Two of the most frequently applied methods are the model predictive control (MPC) and robust control techniques.

Obstacle and collision avoidance is most often solved by applying MPC methods [3, 7, 11, 12, 16]. MPC involves numerical optimisation (occasionally mixed integer programming) at every single time instant and it is a flexible framework, various objectives can be included into the problem formulation. The cost is the increased computational complexity that may require more computational power than what currently exists.

Other approaches include robust control methods [5, 6, 8, 10, 17] that can guarantee certain types of robustness and performance but cannot handle hard constraints the way MPC can. This is the motivation of the method we propose in the following. A promising formation stabilising algorithm is presented in [8], which ensures that vehicles reach a desired formation, even if the communication topology changes arbitrarily and arbitrarily quickly. It utilises the graph theoretical results of [2]. However, it does not guarantee that vehicles do not collide with each other during the transients. We extend this approach by a higher level method effectively which tackles the above problem, even for a relatively large group of vehicles.

Tab. 1. Effects considered in the quadrotor model.

Affected subsystem	Effect	Description
Translation	Gravity	$-mR_t^T G$
	Aerodynamic friction	$-K_t v$
Rotation	Gyroscopic effect	$-\omega \times (I_r \Omega_r)$
	Aerodynamic friction	$-K_r \omega$

The article is structured as follows. Preliminary results are summarised in Section 2, which include the previous results of the authors and present the method, the capabilities of which is extended by our new method. The main contribution of the article, i.e. the safe path generating algorithm is presented in Section 3, which is followed by a practical example in Section 4. The article ends with a short conclusion and summary of the results.

2 Theoretical Fundations

Our control concept consists of three control levels. The internal controller of each quadrotor is a decentralised nonlinear controller using backstepping control. The central controllers are robust distributed formation controllers. A high level controller prevent collision during transients, especially during the change of the communication topology if obstacles appear.

For a single quadrotor we already presented a control method in an earlier paper [9] based on nonlinear backstepping control, hence we summarise here only the dynamic model and the final results of the backstepping control algorithm needed to understand our main concept of formation control. Similarly, we shall use existing results of graph theoretical description of communication topology that will be briefly referenced here.

2.1 Dynamic Model of a Single Quadrotor

Let us assume that a frame (coordinate system) K_E fixed to the Earth can be considered as an inertial frame of reference. The frame fixed to the centre of gravity of the helicopter K_H can be described by its position $\xi = (x, y, z)^T$ and orientation (RPY angles) $\eta = (\Phi, \Theta, \Psi)^T$ relative to K_E. The orientation can be described by the matrix R_t in the following way:

$$R_t = \begin{bmatrix} C_\Theta C_\Psi & S_\Phi S_\Theta C_\Psi - C_\Phi S_\Psi & C_\Phi S_\Theta C_\Psi + S_\Phi S_\Psi \\ C_\Theta S_\Psi & S_\Phi S_\Theta S_\Psi + C_\Phi C_\Psi & C_\Phi S_\Theta S_\Psi - S_\Phi C_\Psi \\ -S_\Theta & S_\Phi C_\Theta & C_\Phi C_\Theta \end{bmatrix}, \qquad (1)$$

where S_x and C_x denote $\sin(x)$ and $\cos(x)$ as usual in robotics. We have assumed that both frames are right-systems and the z-axes are directed upwards.

The relation between $\dot{\xi}$ and $\dot{\eta}$ in K_E and translational and angular velocities v and ω of the helicopter in K_H take the form

$$\dot{\xi} = R_t v, \qquad \omega = R_r \dot{\eta}, \qquad (2)$$

where time derivative is denoted by dot and the matrix R_r has

Tab. 2. Parameters in the dynamic equations.

Property of	Constant	Meaning
Airframe	m	mass of helicopter
	K_t, K_r	aerodynamic coefficients
	I_c	helicopter inertia
	l	distance between CoG and motor axis
Electronics	I_r	rotor inertia
	b, d	force & torque coefficients

the form

$$R_r = \begin{bmatrix} 1 & 0 & -S_\Theta \\ 0 & C_\Phi & S_\Phi C_\Theta \\ 0 & -S_\Phi & C_\Phi C_\Theta \end{bmatrix}. \qquad (3)$$

The helicopter has four actuators (four brushless DC motors), which exert a lift force proportional to the square of the angular velocities Ω_i of the actuators ($f_i = b\Omega_i^2$). The BLDC motors' reference signals can be programmed in Ω_i. The resulting torque and lift force are

$$T = \begin{pmatrix} lb(\Omega_4^2 - \Omega_2^2) \\ lb(\Omega_3^2 - \Omega_1^2) \\ d(\Omega_2^2 + \Omega_4^2 - \Omega_1^2 - \Omega_3^2) \end{pmatrix} \qquad (4)$$
$$F = \begin{pmatrix} 0 & 0 & f \end{pmatrix}^T,$$

where $f = \sum_{i=1}^4 f_i$.

The equations of motion of the helicopter can be obtained by applying the Newton – Euler method:

$$\begin{aligned} F &= mR_t^T \ddot{\xi} + K_t R_t^T \dot{\xi} + mR_t^T G \\ T &= I_c R_r \ddot{\eta} + I_c \left(\frac{\partial R_r}{\partial \Phi} \dot{\Phi} + \frac{\partial R_r}{\partial \Theta} \dot{\Theta} \right) \dot{\eta} + \\ &\quad + K_r R_r \dot{\eta} + (R_r \dot{\eta}) \times (I_c R_r \dot{\eta} + I_r \Omega_r). \end{aligned} \qquad (5)$$

The force and torque components are listed in Tab. 1, while the constants appearing in the equations can be found in Tab. 2.

By neglecting the inductance of BLDC motors, their dynamics can be described as

$$\dot{\Omega}_k = -k_{\Omega,0} - k_{\Omega,1}\Omega_k - k_{\Omega,2}\Omega_k^2 + k_u u_{m,k} \quad k = 1, \ldots, 4, \qquad (6)$$

where the motor parameters are combined into $k_{\Omega,0}$, $k_{\Omega,1}$ and $k_{\Omega,2}$, while the voltage applied to each motor is denoted by $u_{m,k}$.

2.2 Backstepping Control of a Quadrotor

First, we have to reformulate the equations (5) and (6) to make the backstepping algorithm more compact.

$$\begin{aligned} \ddot{\xi} &= f_\xi + g_\xi u_\xi \\ \ddot{\eta} &= f_\eta + g_\eta u_\eta \\ \dot{\Omega}_k &= f_{\Omega,k} + g_{\Omega,k} u_{\Omega,k}, \end{aligned} \qquad (7)$$

where f_ξ, g_ξ and u_ξ are

$$\begin{aligned} f_\xi &= -G - \frac{1}{m} R_t K_t R_t^T \dot{\xi} \\ g_\xi &= \frac{1}{m} diag(r_{t,3}) \\ u_\xi &= (f, f, f)^T. \end{aligned} \tag{8}$$

In the equations above, f_η, g_η and u_η stand for

$$\begin{aligned} f_\eta &= (I_c R_r)^{-1} \left[-I_c \left(\frac{\partial R_r}{\partial \Phi} \dot{\Phi} + \frac{\partial R_r}{\partial \Theta} \dot{\Theta} \right) \dot{\eta} - \right. \\ &\qquad \left. - K_r R_r \dot{\eta} - (R_r \dot{\eta}) \times (I_c R_r \dot{\eta} + I_r \Omega_r) \right] \\ g_\eta &= (I_c R_r)^{-1} \\ u_\eta &= T, \end{aligned} \tag{9}$$

$f_{\Omega,k}$, $g_{\Omega,k}$ and $u_{\Omega,k}$ yield

$$\begin{aligned} f_{\Omega,k} &= -k_{\Omega,0} - k_{\Omega,1} \Omega_k - k_{\Omega,2} \Omega_k^2 \\ g_{\Omega,k} &= k_u \\ u_{\Omega,k} &= u_{m,k}. \end{aligned} \tag{10}$$

Furthermore, the term $r_{t,3}$ appearing in (8) is the third column of R_t.

Since the helicopter is underactuated, the concept is that the helicopter is required to track a path defined by its (x_d, y_d, z_d, Ψ_d) coordinates. The control algorithm can be divided into three main parts. At first, the translational part of the vehicle dynamics is controlled, which then produces the two missing reference signals Φ_d and Θ_d to the attitude control system. The third part is responsible for generating the input signals of the BLDC motors.

The hierarchical structure of the internal controller of a single quadrotor is shown in Fig. 1, where indices d and m denote desired and measured values, respectively. The speed ratio of the three parts of the hierarchical structure depends on the physical properties of the components, especially on the measurement frequency of the sensors. The ideal values of the sampling times for position and orientation control are between 10 – 30 ms. Kalman filters can tolerate the difference of measurement frequencies of the position and orientation (vision system) and acceleration and velocity (IMU). The sampling time of the motor control is set to 10 ms.

2.2.1 Position Control

Let us define the path tracking error

$$q_{\xi_1} = \xi_d - \xi. \tag{11}$$

Applying Lyapunov's theorem we are free to approach

$$\dot{\xi} = \dot{\xi}_d + A_{\xi_1} q_{\xi_1}, \tag{12}$$

where the matrix A_{ξ_1} is positive definite. Introducing a virtual tracking error

$$q_{\xi_2} := \dot{\xi} - \dot{\xi}_d - A_{\xi_1} q_{\xi_1} = -\dot{q}_{\xi_1} - A_{\xi_1} q_{\xi_1} \tag{13}$$

and applying Lyapunov's theory once more we are free to choose

$$\begin{aligned} u_\xi &= g_\xi^{-1} [q_{\xi_1} - f_\xi + \ddot{\xi}_d - A_{\xi_1}(q_{\xi_2} + A_{\xi_1} q_{\xi_1}) - A_{\xi_2} q_{\xi_2}] = \\ &= g_\xi^{-1} [\ddot{\xi}_d - f_\xi + (I_3 + A_{\xi_2} A_{\xi_1}) q_{\xi_1} + (A_{\xi_2} + A_{\xi_1}) \dot{q}_{\xi_1}], \end{aligned} \tag{14}$$

where A_{ξ_2} is positive definite. The resulting system is

$$\ddot{\xi} = \ddot{\xi}_d + (I_3 + A_{\xi_2} A_{\xi_1}) q_{\xi_1} + (A_{\xi_2} + A_{\xi_1}) \dot{q}_{\xi_1}. \tag{15}$$

Then the Lyapunov function of the closed loop system satisfies

$$V(q_{\xi_1}, q_{\xi_2}) = \frac{1}{2} \left(q_{\xi_1}^T q_{\xi_1} + q_{\xi_2}^T q_{\xi_2} \right) > 0 \tag{16}$$

$$\dot{V}(q_{\xi_1}, q_{\xi_2}) = -q_{\xi_1}^T A_{\xi_1} q_{\xi_1} - q_{\xi_2}^T A_{\xi_2} q_{\xi_2} < 0 \tag{17}$$

so that stability is guaranteed and equivalent to

$$0 = \ddot{q}_{\xi_1} + (A_{\xi_2} + A_{\xi_1}) \dot{q}_{\xi_1} + (I_3 + A_{\xi_2} A_{\xi_1}) q_{\xi_1}. \tag{18}$$

Assuming positive definite and diagonal A_{ξ_1}, A_{ξ_2} matrices with diagonal elements $a_{\xi_1,i}$, $a_{\xi_2,i}$, the characteristic equations have the form

$$s^2 + (a_{\xi_2,i} + a_{\xi_1,i}) s + (1 + a_{\xi_2,i} a_{\xi_1,i}) = 0, \tag{19}$$

which guarantees stability.

Furthermore, if the term involving the second derivative of the reference can be kept small compared to the others ($\ddot{\xi}_d \approx 0$), the transfer function from each reference component to the corresponding output takes the form

$$P_{\xi_i, \xi_{d,i}}(s) = \frac{(a_{\xi,2,i} + a_{\xi,1,i}) s + (1 + a_{\xi,2,i} a_{\xi,1,i})}{s^2 + (a_{\xi,2,i} + a_{\xi,1,i}) s + (1 + a_{\xi,2,i} a_{\xi,1,i})}, \tag{20}$$

where the constants $a_{\xi,\bullet,i}$ are the ith diagonal elements of $A_{\xi,\bullet}$. These transfer functions can later be utilised when designing the higher level control algorithm.

This means that the errors exponentially converge to zero if the calculated values of f_ξ and g_ξ are close to the real ones. Algebraic manipulations can be performed in $g_\xi u_\xi$. The third component of u_ξ is the lift force f. Since the entire controlled system is stable, g_ξ has to be convergent and its limit is $(0, 0, 1)^T/m$, hence the reference signals Φ_d and Θ_d can be obtained as follows.

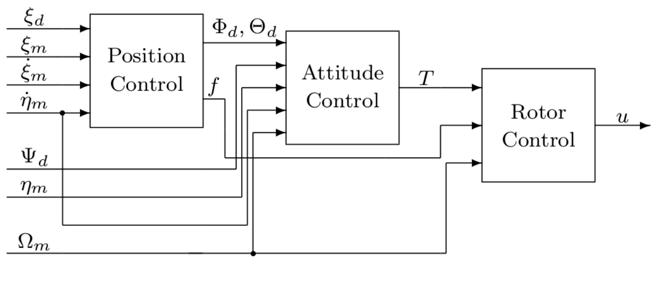


Fig. 1. The hierarchical structure of the internal controller of a single quadrotor.

First, using the state variables Φ, Θ, Ψ, we compute

$$\tilde{u}_\xi = \begin{pmatrix} C_\Phi S_\Theta C_\Psi + S_\Phi S_\Psi \\ C_\Phi S_\Theta S_\Psi - S_\Phi C_\Psi \\ f \end{pmatrix} =: \begin{pmatrix} u_{\xi_x} \\ u_{\xi_y} \\ f \end{pmatrix}. \tag{21}$$

Then, from u_{ξ_x} and u_{ξ_y} we determine the reference signals for the attitude control:

$$\begin{aligned} S_{\Phi_d} &= S_\Psi u_{\xi_x} - C_\Psi u_{\xi_y} \\ S_{\Theta_d} &= \frac{C_\Psi u_{\xi_x} + S_\Psi u_{\xi_y}}{C_\Phi} \end{aligned} \tag{22}$$

The reason why these signals can be considered as reference signals is that as the helicopter approaches the desired coordinates, they converge to zero. Conversely, if the helicopter follows the appropriate attitude and lift force, it will reach the desired position and orientation.

The control law of the attitude subsystem is obtained in a similar fashion. For details, see [9].

2.2.2 Rotor Control

Since the rotor equations are of first order, there is no need for the virtual error q_{m_2}. However, it is worth including the derivative of q_{m_1} similarly as in the previous sections because of the error dynamics:

$$\begin{aligned} u_m = g_m^{-1}[\dot{\Omega}_d - f_m + (I_4 + A_{m_2}A_{m_1})q_{m_1} + \\ + (A_{m_2} + A_{m_1})\dot{q}_{m_1}], \end{aligned} \tag{23}$$

with q_{m_1} and f_m being

$$q_{m_1} = \begin{pmatrix} \Omega_{1_d} - \Omega_1 \\ \Omega_{2_d} - \Omega_2 \\ \Omega_{3_d} - \Omega_3 \\ \Omega_{4_d} - \Omega_4 \end{pmatrix} \quad \text{and} \quad f_m = \begin{pmatrix} f_{m,1} \\ f_{m,2} \\ f_{m,3} \\ f_{m,4} \end{pmatrix}. \tag{24}$$

Since the four motors are considered to be identical, g_m can be any of $g_{m,k}$-s and therefore it is a scalar. It is worth noticing that since T and f are linear combinations of Ω_k^2, Ω_{k_d} are the element-wise square roots of

$$\begin{pmatrix} \Omega_{1_d}^2 \\ \Omega_{2_d}^2 \\ \Omega_{3_d}^2 \\ \Omega_{4_d}^2 \end{pmatrix} = \begin{bmatrix} 0 & -(2lb)^{-1} & -(4d)^{-1} & (4d)^{-1} \\ -(2lb)^{-1} & 0 & (4d)^{-1} & (4d)^{-1} \\ 0 & (2lb)^{-1} & -(4d)^{-1} & (4d)^{-1} \\ (2lb)^{-1} & 0 & (4d)^{-1} & (4d)^{-1} \end{bmatrix} \begin{pmatrix} T \\ f \end{pmatrix}. \tag{25}$$

For stability reasons, A_{m_1} and A_{m_2} should be positive definite matrices.

2.3 Formation Stability of Linear Systems in Graph Theoretical Approach

The relation between formation stability of connected linear systems and graph-theory was discussed in the pioneering work of Fax and Murray [2]. Here we summarise some of the main results needed for our purposes.

2.3.1 Normalised Laplacian Matrix of the Communication Topology

Let us consider the formation of N discrete time linear systems communicating each other. The communication topology is defined by a directed graph. In the graph the vertices are the linear systems and the edges indicate the communication links. The arrow of the edge points to the receiver. Denote $\mathcal{J}_i$ the set of neighbours from which node i receives information and let $|\mathcal{J}_i|$ be its cardinality. We assume normalised Laplacian matrix L of type $N \times N$ defined as

$$L_{ik} = \begin{cases} 1, & k = i \\ -\dfrac{1}{|\mathcal{J}_i|}, & k \in \mathcal{J}_i \\ 0, & k \notin \mathcal{J}_i \end{cases} \tag{26}$$

Important properties are as follows [2]:

1) One eigenvalue of L is always zero and the corresponding right eigenvector is **1** (all ones).
2) All eigenvalues λ_i of L lie in the unit disk (Perron disk) centred at $1 + j0$ which means that $\lambda_i = 1 + \delta_i$ where $|\delta_i| \le 1$.
3) If L is undirected, then L has only real eigenvalues.

2.3.2 Closed Loop Formation Stability

One measure of the ith system in the formation may be the equally weighted sum of errors of the sensed neighbours:

$$e_i = \frac{1}{|\mathcal{J}_i|} \sum_{k \in \mathcal{J}_i} e_{ik}. \tag{27}$$

Here e_{ik} describes the error between the ith and kth unit according to

$$e_{ik} = (r_i - v_i) - (r_k - v_k) = \tilde{r}_{ik} - (v_i - v_k), \tag{28}$$

where $\tilde{r}_{ik}$ is the desired relative pose in the formation and v_i, v_k are the transmitted outputs of the vehicles i and k, respectively.

We shall assume that system i has input e_i and output v_i and its transfer function is $H(s)$. Its formation level controller $K_F(s)$ has input e_i and output u_i where e_i is the global formation error information about the error of the unit in the formation (outer loop). A single vehicle with its backstepping controller can be characterised by $T_{v_i,u_i} =: P(s)$ such that $v_i = P(s)u_i$. A vehicle with its local controllers is depicted in Fig. 2.

The communication structure can be described by the Kronecker product $L_{(p)} = L \otimes I_p$ where p is the number of outputs v_i of a single unit. With $\boldsymbol{r} = (r_1, \dots, r_N)^T$, $\boldsymbol{v} = (v_1, \dots, v_N)^T$, $\boldsymbol{e} = (e_1, \dots, e_N)^T$ yields

$$\boldsymbol{e} = L_{(p)}(\boldsymbol{r} - \boldsymbol{v}). \tag{29}$$

Let the dynamics of the ith linear system $P(s)$ be given as

$$\begin{aligned} \dot{x}_i &= Ax_i + Bu_i \\ v_i &= Cx_i. \end{aligned} \tag{30}$$

The following theorem is from Fax and Murray [2].

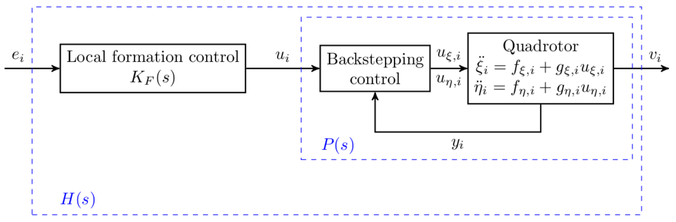

e_i
Local formation control $K_F(s)$
u_i
Backstepping control
$u_{\xi,i}$
$u_{\eta,i}$
Quadrotor
$\ddot{\xi}_i = f_{\xi,i} + g_{\xi,i}u_{\xi,i}$
$\ddot{\eta}_i = f_{\eta,i} + g_{\eta,i}u_{\eta,i}$
v_i
$P(s)$
y_i
$H(s)$

Fig. 2. Single quadrotor with local controllers.

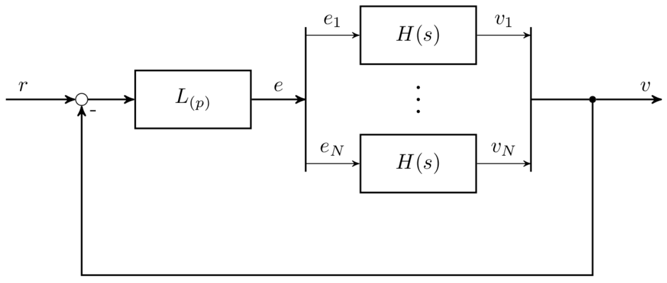


Fig. 3. Closed loop formation.

Theorem 1 *A controller $K_F(s)$ stabilises the closed loop formation (Fig. 3) if and only if it simultaneously stabilises the following set of N systems:*

$$\begin{aligned} \dot{x}_i &= Ax_i + Bu_i \\ v_i &= \lambda_i C x_i \end{aligned} \qquad i = 1,\ldots,N, \tag{31}$$

where $u_i = K_F(s)e_i$, $e_i = -v_i$ and λ_i are the eigenvalues of L.

Notice that λ_i may be complex, hence the theorem requires the stability of systems with complex parameters.

Since the eigenvalues of L lie in the Perron disk, $\lambda_i = 1 + \delta_i$ satisfies $|\delta_i| \leq 1$ where δ_i is complex number and $v_i = Cx_i + \delta_i C x_i$, $i = 1,\ldots,N$.

Using singular value decomposition, C can be factorised as $C = D_\delta C_\delta$. Thus, signals $z_i = C_\delta x_i$ and $w_i = \delta I_q z$ can be introduced, the second playing the role of uncertainty description, where q is the rank of C and $\|\delta\| \leq 1$ [8].

For stability investigation the N units can be covered as a single one with uncertainty $\|\delta\| \leq 1$ and a transfer function $T_{z,w} =: G(s)$ as follows:

$$\begin{aligned} \dot{x} &= Ax + Bu \\ z &= C_\delta x \\ v &= Cx + D_\delta w \\ u &= -K_F v. \end{aligned} \tag{32}$$

Notice that the robust control problem is similar for each component, hence the index has been omitted. The resulting system is $\mathcal{F}_l(G(s), K_F(s))$ where $\mathcal{F}_l$ is the linear fractional transformation from w to z.

The following theorem is from Popov and Werner [8].

Theorem 2 *The controller $K_F(s)$ stabilises the closed loop formation for any number of agents N and under fixed, as well as any time-varying communication topology if the $\mathcal{H}_\infty$ norm of the system $\mathcal{F}_l(G(s), K_F(s))$ is smaller than 1 (Fig. 4(a)).*

The properties of the distributed formation control can be sharpened by using sensitivity $W_S(s)$ and control sensitivity $W_K(s)$ filters (Fig. 5). The interconnection with the controller and the augmented plant can be seen in Fig. 4(b).

3 Safe Formation Change

The most crucial strengths of the algorithms presented above are that they are capable of stabilising a group of any number of vehicles with any kind of communication topology that holds certain connectivity properties. However, there is a major drawback that is not explicitly tackled by the algorithm, i.e. it is not guaranteed that during the transients the vehicles keep safe distance from each other. Linear robust control methods cannot satisfy such constraints. Therefore, either different control algorithms are required for such problems, such as model predictive control (MPC), or collision avoidance must be implemented on a higher level.

The proposed method follows the latter approach. Given a number of identical vehicles in an initial formation (defined by spatial ponts $p_{0,i}$), the task is to occupy the specified target positions $p_{1,i}$ within finite time and keeping a predefined minimum distance between each other during the transition. The vehicles are not assigned a specific target position, the proposed algorithm decides which target is reached by which vehicle. The vehicles track straight paths between the start and target positions and may not necessarily move all at the same time since

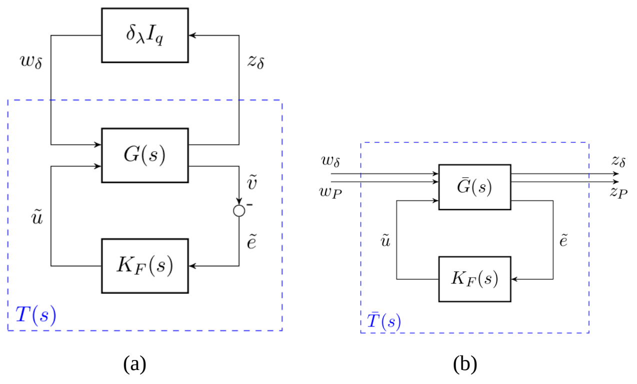


Fig. 4. Plant interconnection.

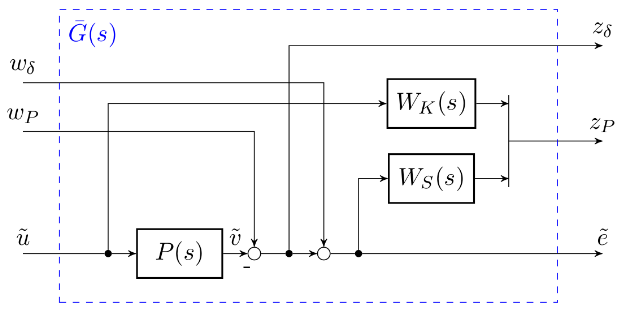


Fig. 5. $\mathcal{H}_\infty$ design setup.

Tab. 3. Algoritm overview.

Phase 1		**Direct transition**
	a)	Selecting candidate paths
	b)	Conflict search in "dual" graph
	c)	Maximum clique search
Phase 2		**Correction routes**
	a)	Selecting candidate correction routes
	b)	Conflict search in "dual" graph
	c)	Maximum clique search
Phase 3		**Resolving trapped targets**
	a)	Checking all routes for conflicts
	b)	Maximum clique search

one might act as an obstacle to the other depending on the structure of the initial and target formation. The algorithm should also take into account that the vehicles have a maximum travelling speed. There is only one restriction, which is related to the formation and the predefined safety distance. The ratio between the minimum distance between each pair of vehicles in their initial and target positions and the safety distance should exceed a constant value specified later:

$$\min_{\substack{i,j \\ i\neq j}} \frac{d_{0,i,j}}{d_s} > c \qquad \min_{\substack{i,j \\ i\neq j}} \frac{d_{1,i,j}}{d_s} > c, \tag{33}$$

where $d_{\bullet,i,j} = \|p_{\bullet,i} - p_{\bullet,j}\|$ and d_s is the safety distance. The crucial aim is to find the smallest possible c. As will be revealed later, the above constraint is not overly restrictive in real applications since the safety distance is related to the physical dimensions of the vehicles.

In the following, the safe path generating method will be presented, then as an illustration, a formation changing scenario will be shown.

3.1 Path Generating Algorithm

The basic idea of the proposed algorithm is to avoid the online path planning and optimisation at every sample time instant. Instead, only if the formation of the vehicle group has to be changed, safe trajectories will be generated in a simple but efficient way. The generated paths will be safe at the same time. The steps of the method are described in Tab. 3 and the integration into the formation is shown in Fig. 6. The first two phases may consist of several steps. During the first phase, as many vehicles as possible move directly from their initial positions to certain target positions. In the second phase, certain vehicles that have already reached a target regroup so that empty targets are generated in the proximity of new vehicles. In the last phase, vehicles that remain in their initial positions can simultaneously move to a target.

The key in each phase is how to determine which vehicles are allowed to move at the same time. Graphs will be constructed that contain information about the risk of collision. The number of vehicles taking part in each step will correspond to the size of a clique in this graph. For computational reasons, certain heuristics will also be included in the algorithm. The steps that have to be carried out throughout the path generation are as follows.

3.1.1 Phase 1 – Direct Transition

First, a graph $\mathcal{N}$ describing the candidate routes has to be formed. The vertices of the graph correspond to the initial and target positions and the edges correspond to a route between an initial and a target point. Since in the simplest case every vehicle has the possibility of travelling towards any target point, this graph is a full bipartite graph (see Fig. 7).

Next, it should be checked whether a route conflicts with another. In this context, two routes are in *conflict* with each other when the distance between them is less than the safety distance d_s. (This definition is obviously conservative in the sense that it does not take into account the motion of the vehicles, only their paths.) These pieces of information can be collected into a "dual" graph $\mathcal{M}$ where each vertex corresponds to an edge in $\mathcal{N}$ (marked by green in Fig. 7) and there is an edge between two vertices if the distance between the corresponding two routes is greater than d_s.

The task is then to find as many routes as possible among which there do not exist pairs that are in conflict with each other. In other words, a maximum clique has to be found within $\mathcal{G}(\mathcal{M})$, which is the adjacency graph of $\mathcal{M}$.

It is known that the maximum clique cannot contain more vertices than the number of vehicles. However, in most cases the size of the maximum clique is less than this value, due to the fact that vehicles can act as "obstacles" to each other. Therefore, the above method has to be repeated as long as there are new vehicles that can find their way to the targets. In certain cases, this algorithm cannot guarantee that all the vehicles reach a target position. Therefore, a variant of this method has to be applied then, which further reduces the number of vehicles that cannot reach a target point.

3.1.2 Phase 2 – Correction Routes

For this purpose, the notion of correction route has to be introduced. A *correction route* connects an occupied initial position with an unoccupied target position via intermediate occupied positions. Each section consists of straight paths. The intermediate points are not necessarily target points. However,

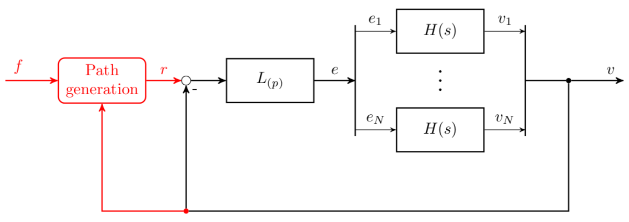


Fig. 6. Formation change logic.

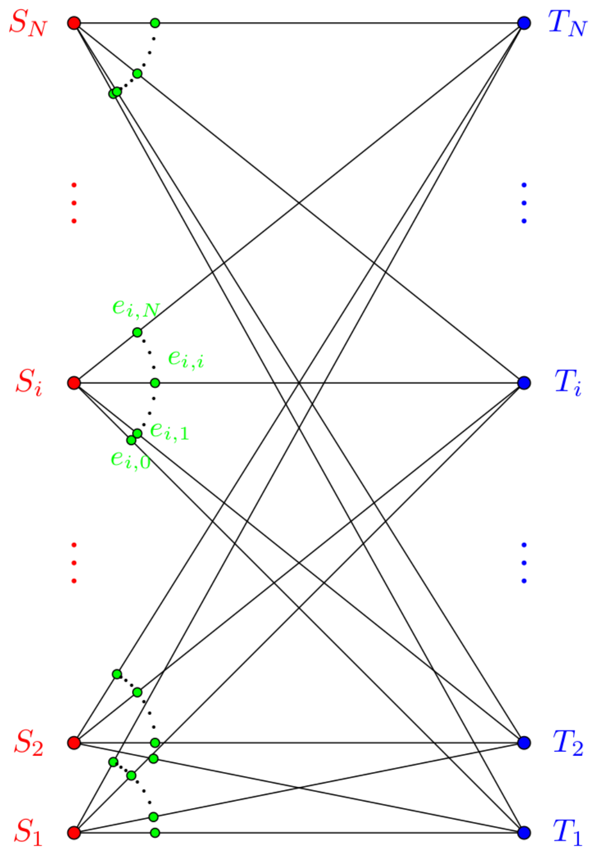


Fig. 7. Path search graph.

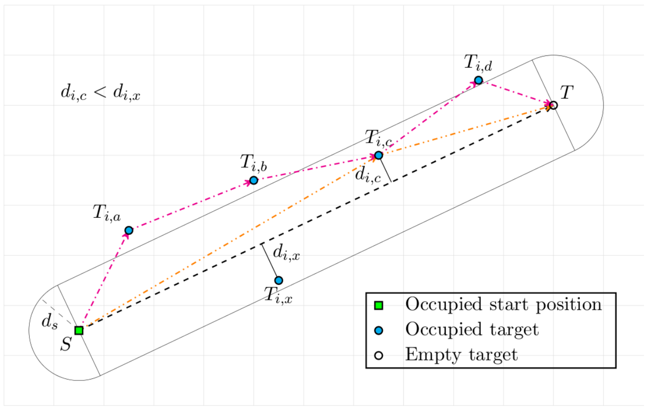


Fig. 8. Correction route generation.

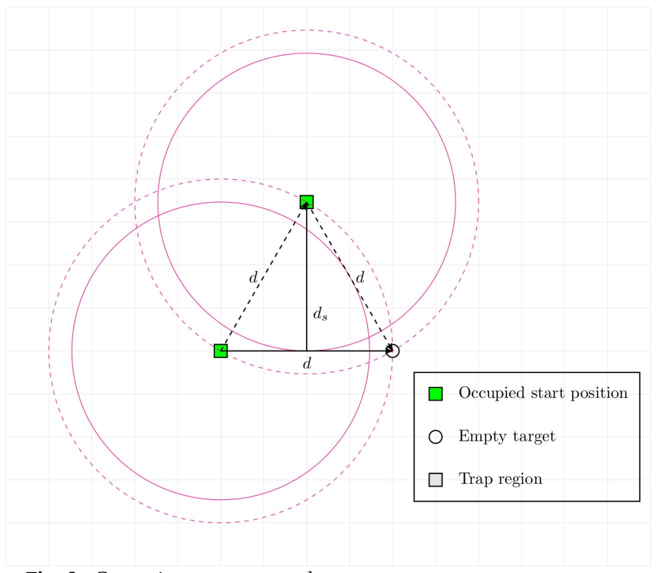


Fig. 9. Correction route extremal case.

if a correction route involves an occupied initial point, then there exists a shorter correction route between this intermediate point and the same target point. Therefore, only the shortest correction routes are considered in the following. The iterative process consists of the following steps. The first task is to find an occupied intermediate point P_i with minimum distance from the line section between the current start and target position (initially $\overrightarrow{ST}$) within the safety distance. If no such point is found, the route is generated. Otherwise, correction route generation is split into two parts. Finally, when route generation is finished, the intermediate points have to be collected in the right order. Correction route generation is illustrated in Fig. 8. The first intermediate target point found during the process is $T_{i,c}$ since the other candidate $T_{i,x}$ is farther from $\overrightarrow{ST}$.

When searching for correction routes, it has to be ensured that each intermediate point is closer to the target point than the previous one including the starting point. Otherwise, correction routes could possibly be infinite. Another aspect that has to be taken into account is that each correction route should be as short as possible in order to avoid unnecessary time and energy consumption.

The purpose of correction routes is that along the segments of each such route the vehicles can regroup creating an unoccupied target point that can be reached by a new vehicle. This regrouping can be performed either sequentially backwards from the target or in parallel. The latter is beneficial since it reduces the total time and energy required for the change of formation. However, correction routes need to have the property that none of the vehicles travelling from one intermediate point to the subsequent reach each other within the safety distance. This makes the search computationally slightly more involved.

If correction routes that satisfy the above requirements exist, another search, similar to the direct transition step can be performed. The only difference is that it has to be defined what *conflict* means between a pair of correction routes. This depends on whether the vehicles move along the correction routes sequentially or in parallel. In the former case, conflict between subsequent pairs of route sections have to be checked, while in the other none of the line segments of one correction route can be in conflict with any in the other so that the whole correction routes are not in conflict with each other.

Apart from certain exceptions, it can be proved geometrically that if

$$d_{i,j} > \frac{2}{\sqrt{3}} d_s \tag{34}$$

holds, then all the intermediate points in all the correction routes are closer to the targets than the previous ones and the initial points (see Fig. 9). This corresponds to $c_{\min} = 2/\sqrt{3}$.

3.1.3 Phase 3 - Trapped Targets

One of the two problems that has to be tackled is that certain configurations are still complicated or even impossible to resolve. This is the case when an unoccupied target is *trapped* by two or more occupied start positions. This means it is within the safety distance from all the involved vehicles, therefore more complex manoeuvres, possibly including even more vehicles, are required than those primarily involved in the trap (illustrated in Fig. 10). What makes the problem even more serious, it might occur that fewer start positions are involved in a trap than empty targets. This means there are other vehicles that cannot reach any target, though they are not involved in forming traps. The second problem is that the first segment of every correction route connects a start and a target position and the above condition does not necessarily hold.

The first problem can be tackled by increasing the rather strict constraint $c_{\min} = 2/\sqrt{3}$ to $c_{\min} = \sqrt{2}$, it can be ensured that all the trapped positions can be reached simultaneously in one step, which corresponds to a maximum clique of a size of the remaining vehicles in $\mathcal{M}$ of the very last step of the algorithm. The reason for this is the following. In case $c_{\min} = \sqrt{2}$, the greatest distance between two points within the intersection of two start positions is not greater than $c_{\min} d_s$, see Fig. 10. Each intersection can thus contain either 0 or 1 target. As a consequence, trapped positions can only occur in closed chains or closed three-dimensional surfaces and in case vehicles travel at constant speeds along straight lines they never get within the safety distance from each other. The change of the distance ratio has no real restrictive effects considering that the original one meant that the densest possible configuration was the same as placing as many spheres in a certain volume as possible, while the latter one corresponds to a cubic grid of the same edge length.

3.1.4 Generation of Suitable Correction Routes

The other problem mentioned above is illustrated in Fig. 11. Suppose a correction route has to be generated from start position S and target T. When generating the correction route, vehicles may have already occupied target positions in the red area, which is within the safety region of route $S \to T$. The distance between a vehicle in the red area and the target is greater than $\|\overrightarrow{ST}\|$. Since these points cause divergence from the target, it should be avoided that correction routes include them as intermediate points.

A solution to this problem is as follows. If all the routes that end in a target point which has an initial point within an increased safety distance d'_s are filtered out, then it is ensured that suitable correction routes can be found in each step. The ratio between d'_s and d_s can be read from the figure when $d = c \cdot d_s$:

$$d'_s = d_s \cdot c \sqrt{2\left(1 - \sqrt{1 - \frac{1}{c^2}}\right)}. \tag{35}$$

The downside, however, is that $c_{\min}$ has to be increased by

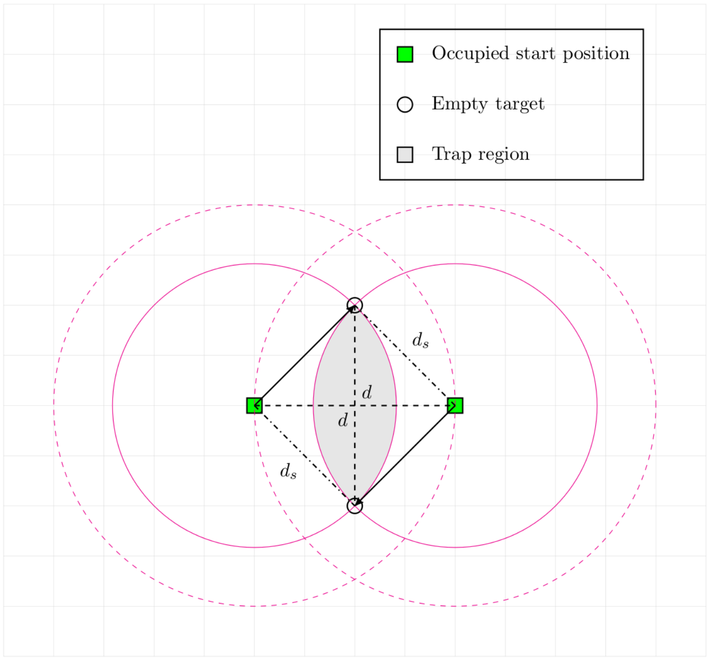
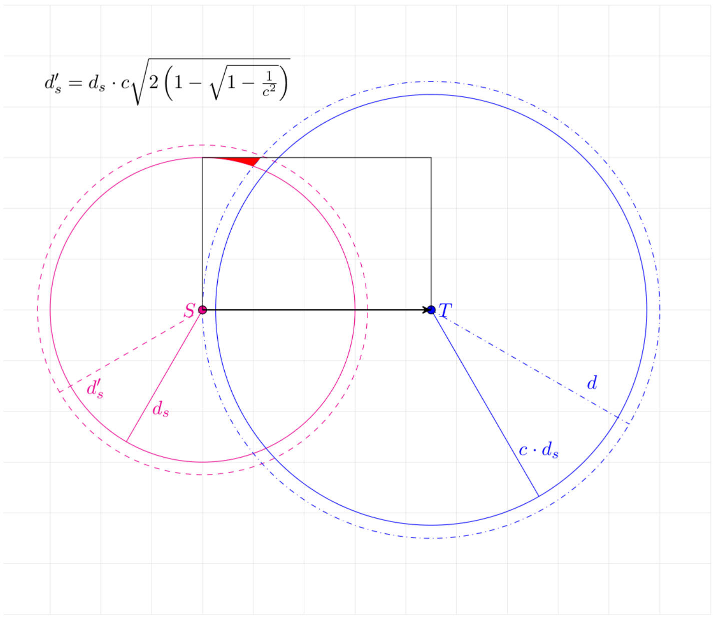


Fig. 10. Trapped vehicles (extremal case, $c = \sqrt{2}$).



$d'_s = d_s \cdot c\sqrt{2\left(1-\sqrt{1-\frac{1}{c^2}}\right)}$
S
T
d'_s
d_s
d
$c \cdot d_s$

Fig. 11. Ensuring convergence to the target.

the same ratio, as it is revealed in the configuration depicted in Fig. 12. A vehicle in the red region in Fig. 11 may block vehicles from reaching targets. If these points are kept empty, they may act as if they were trapped, thus they are treated as trapped. Therefore, the ratio between d and d'_s should be kept at $\sqrt{2}$, which yield $c_{\min} = 4/\sqrt{7}$. It has to be mentioned that the change is less than 7%, which is not an overly strict constraint.

It also has to be mentioned that in case $c > \sqrt{2}$, every vehicle in a correction route can move at the same time without the risk of collision, apart from the vehicle in the start position. It has to be checked separately whether there is a risk of collision with the next vehicle or not.

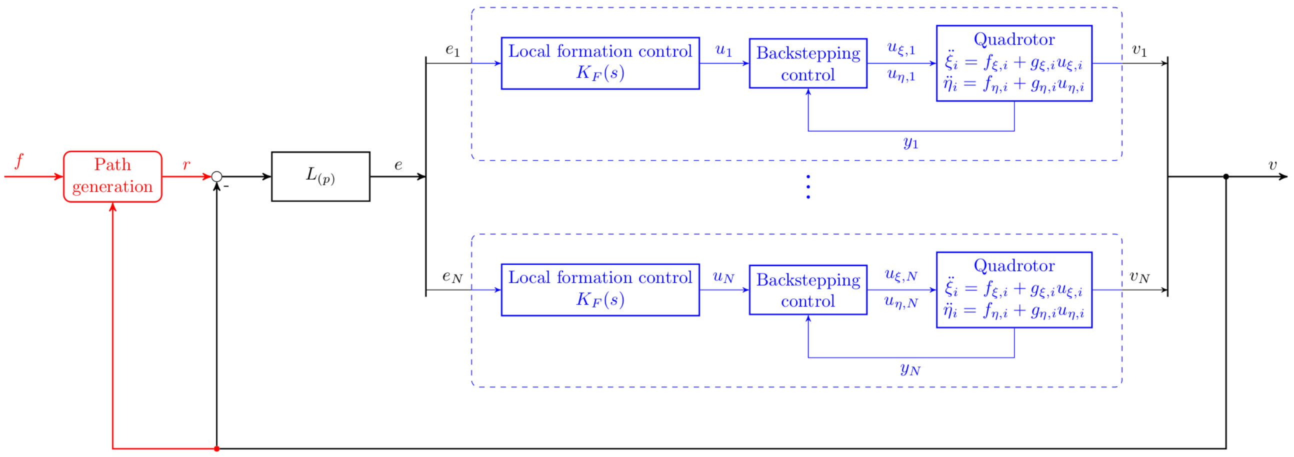

f
Path generation
r
-
$L_{(p)}$
e
e_1
Local formation control $K_F(s)$
u_1
Backstepping control
$u_{\xi,1}$
$u_{\eta,1}$
Quadrotor
$\ddot{\xi}_i = f_{\xi,i} + g_{\xi,i}u_{\xi,i}$
$\ddot{\eta}_i = f_{\eta,i} + g_{\eta,i}u_{\eta,i}$
v_1
y_1
e_N
Local formation control $K_F(s)$
u_N
Backstepping control
$u_{\xi,N}$
$u_{\eta,N}$
Quadrotor
$\ddot{\xi}_i = f_{\xi,i} + g_{\xi,i}u_{\xi,i}$
$\ddot{\eta}_i = f_{\eta,i} + g_{\eta,i}u_{\eta,i}$
v_N
y_N
v

Fig. 13. Simulation setup.

3.2 Clique Finding in $\mathcal{G}(\mathcal{M})$

A number of maximum clique search algorithms have already been developed by research groups, see e.g. [4, 13–15, 18–20]. The algorithm presented in [4] is considered as an efficient method in most cases, thus it is applied to our problem as well. In general, maximum clique algorithms consist of a graph colouring step, which is a computationally hard problem. Therefore, they all utilise some kind of heuristics. Graph colouring aids the selection of new candidate vertices that may increase the size of the currently growing clique. The other part of these algorithms is a continuous test whether the new candidate vertex and the growing clique form a clique together or not.

Since finding a maximum clique in a graph is known to be NP-complete [1], certain modifications are necessary to be applied to the algorithm so that it is tractable in case the number of vehicles reaches the order of 50. One way of accelerating the search is that during the graph construction step, only a subset of all possible routes are considered. There are a number of ways of selecting these routes:

1) Taking only the *n* closest targets to each initial position,
2) Taking only the *n* closest initial positions to each target,
3) Combining the first two methods,
4) Sorting the target distances from each initial position and selecting *n* evenly.

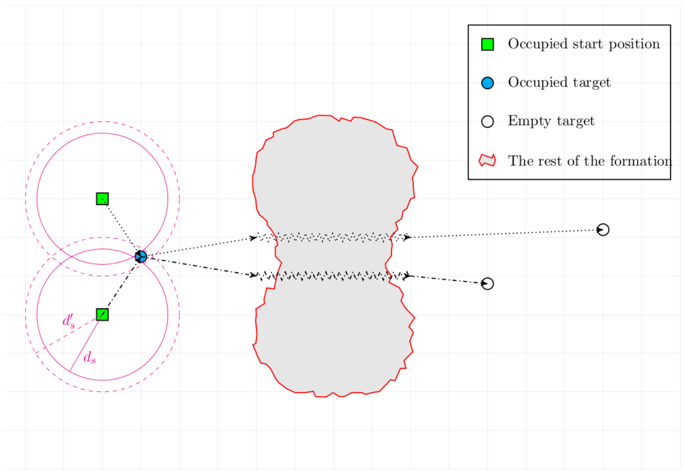


Fig. 12. Difficulty caused by vehicles in the red zone in Fig. 11.

The last in the list performs the best in most cases as the others tend to focus on different groups of vehicles. Note that this step is also important since considerable time is required for creating the adjacency matrix itself, since its original size is N^2-by-N^2!

Even though this modification greatly reduces the search space, finding the maximum clique in the reduced graph may still require a long time. A possible method is to limit the total search time. Another tweak is an experimental one. In most practical cases a first candidate clique is found in a short time, the size of which is not much less than that of the maximum clique. Finding new candidates can be time consuming. Thus, a time limit is introduced that sets a maximum time between every new candidate cliques.

The above modifications are destructive in the sense that applying them most likely results in finding a clique whose size is less than that of the maximum clique of the original adjacency matrix. However, all the vehicles still reach a target point, though the number of iterations may increase.

The next type of modification has only slight impact on the size of the clique found. It rather aims at finding cliques that involve the shortest path possible. The purpose is obviously that time and thus energy consumption should be kept as low as possible, even though this is not the major objective of the whole process. This can be done by a simple tweak. Route lengths are already available when the clique search begins. Therefore, these pieces of information can be utilised as a tie-breaker when

Tab. 4. Backstepping control parameters.

Subsystem	a_1	a_2
x, y	2.0	1.5
z	2.5	1.5
Ψ	15.0	10.0

sorting the vertices based on their degree (c.f. lines 9 – 13 of Fig. 4 in [4]). This way, the shortest routes are checked as early as possible.

4 Formation Change Scenario

As an illustrative example, a formation change manoeuvre involving a group of 25 quadrotors is presented. The vehicles are placed randomly in the 3D space and the target positions are also chosen randomly in the *xy*-plane, satisfying the constraints of (33) with the constant $c = 4/\sqrt{7}$. The vehicles point to the same direction ($\Psi_{d,i} = 0$) throughout the mission.

Simulation is performed according to Fig. 13. Communication topology is chosen randomly, two vehicles are connected with a probability of 0.2, which means that each vehicle exchanges information with 5 others on average. For simplicity, the topology is fixed throughout the mission.

The coefficients of the backstepping controller can be seen in Tab. 4. The resulting transfer functions take similar form to (20).

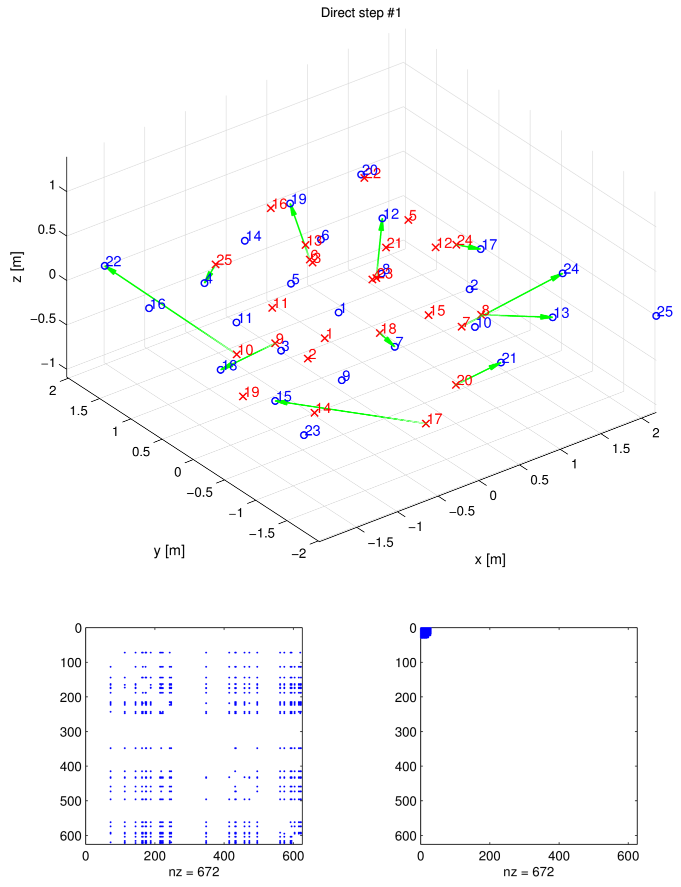


Fig. 14. Example scenario, direct phase, step 1.

Tab. 5. Path generation statistics.

Phase	Step #	$t_{\mathcal{G}}$ [s]	t_{MC}[s]	$\lvert\mathrm{MC}\rvert$
Direct	1	0.8356	0.0288	11
	2	0.3865	0.0241	8
	3	0.0650	0.0024	4
Correction	1	0.0261	0.0007	2

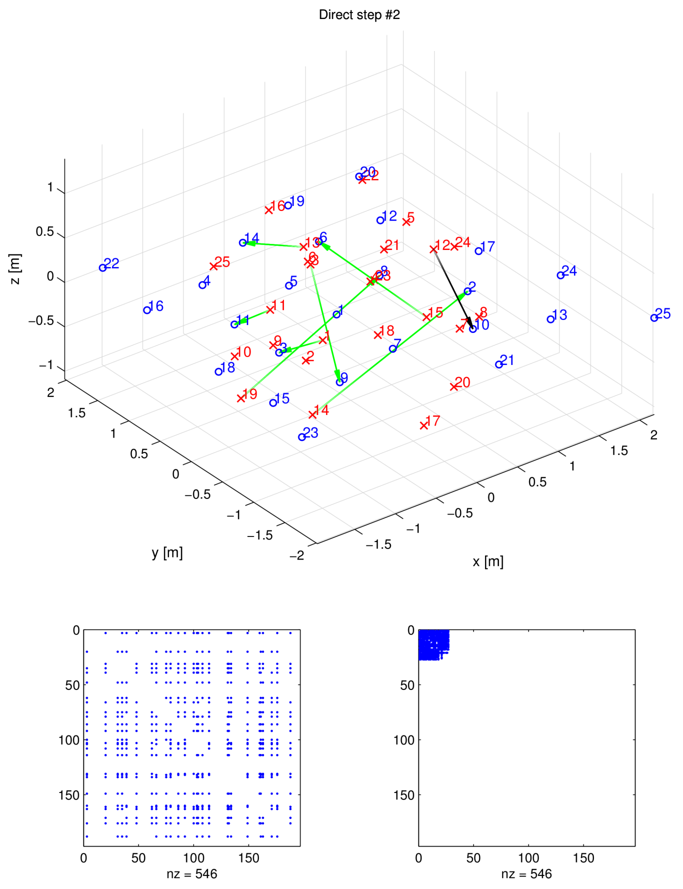


Fig. 15. Example scenario, direct phase, step 2.

The formation controller parameters are obtained by setting the weighting functions in Fig. 5 to

$$
\begin{aligned}
W_{S,\xi_i} &= 10^7 \left(\frac{\frac{1}{7.5} s + 1}{\frac{1}{3.75\cdot 10^{-4}} s + 1} \right)^2 \\
W_{K,\xi_i} &= 3 \cdot 10^{-2} \left(\frac{\frac{1}{7.5\cdot 10^{1}} s + 1}{\frac{1}{3\cdot 10^{2}} s + 1} \right)^3 \\
W_{S,\Psi} &= 10^2 \frac{\frac{1}{10} s + 1}{\frac{1}{10^{-3}} s + 1} \\
W_{K,\Psi} &= 3 \cdot 10^{-1} \left(\frac{\frac{1}{2} s + 1}{\frac{1}{2\cdot 10^{2}} s + 1} \right)^3 .
\end{aligned}
\qquad (36)
$$

The transitional dynamics have to satisfy more stringent constraints. Since formation change involves steady linear motion, vehicles should diverge from the path as little as possible. Thus, instead of the commonly applied routine, the difference from a ramp input is penalised, which corresponds to the increased gain at lower frequencies in the weighting function W_{S,ξ_i}. Moreover, this method ensures that it is not necessary to split up the motion into accelerating, travelling and decelerating parts. Acceleration is bounded by the aid of W_{K,ξ_i}.

Performance requirements for the rotation about the *z*-axis are less stringent and thus the order of the controller is lower (6, compared to 7 in the case of translational motion). Robust stability with desired performance are achieved in both cases and all the designed controllers are stable. The full formation-level controller is obtained by placing the four controllers in the diagonal of a 4-by-4 matrix.

Reference paths were generated so that the speed of vehicles never exceeds 1 m/s. Such setting is necessary for guaranteeing the stability of the backstepping controller of each vehicle. Reference paths in each formation change step are designed so that vehicles involved in the current step start moving and reach target at the same time. Computation time statistics are shown in Tab. 5, where columns $t_{\mathcal{G}}$, t_{MC} and $|MC|$ show the time required for adjacency matrix generation, finding a maximum clique and the clique's size, respectively. Tests were performed using MATLAB on an average P4 PC. All the algorithms were executed on a single core. It can be seen that the most time consuming step is the first, in particular the adjacency graph generation, which is common in general situations.

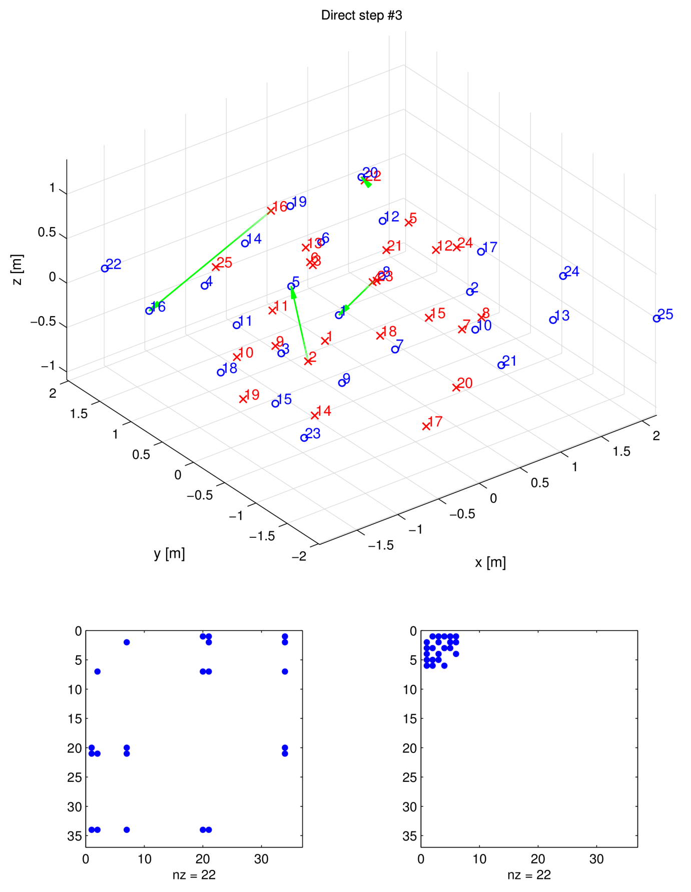


Fig. 16. Example scenario, direct phase, step 3.

The steps of the example formation change are shown in Figs. 14, 15, 16 and 17. Each figure consists of two main parts. The upper graph shows the paths of vehicles involved in the transition step. Start and target positions are marked red crosses and blue circles, respectively. Only vehicles that change position are shown for transparency reasons. An additional dashed arrow connects the starting and end point of each correction route in the figures presenting the correction steps. Black arrows show the motion of vehicle 12 (the one which starts from initial position 12 and reaches target point 13 via target point 10). In the lower part, the sparsity pattern of $\mathcal{G}(\mathcal{M})$ is shown. The left pattern is the original one, the right pattern is obtained by sorting the rows and columns based on the degree of each vertex. The number of nonzero elements of the matrices are also shown. At each step, a maximum of 5 of all the possible routes are selected from each occupied start position. Conflicts are checked only among these routes, in order to accelerate path generation. It is worth mentioning that trapped targets occur rarely in practice since vehicles that might be involved in such situations usually find their way to different target points.

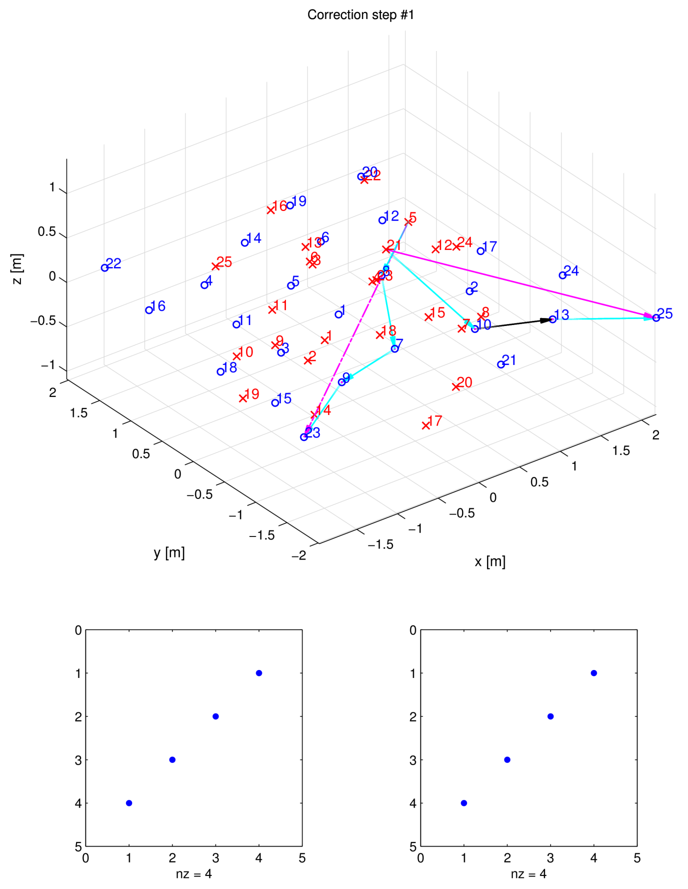


Fig. 17. Example scenario, correction phase, step 1.

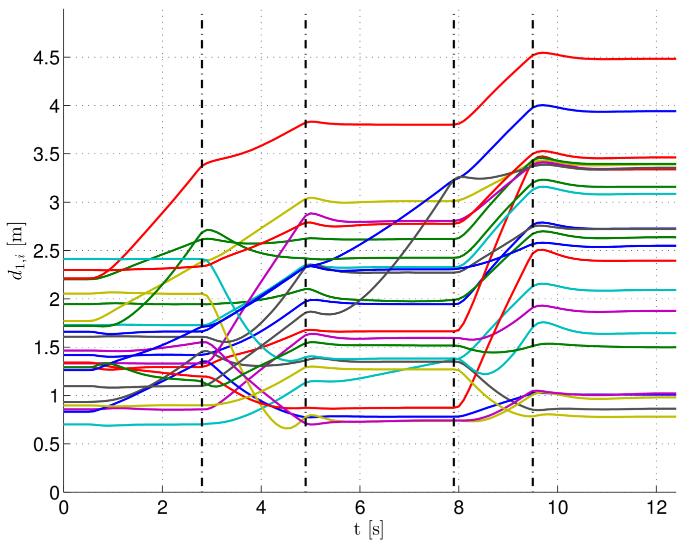


Fig. 18. Distance between vehicle 12 and the other vehicles.

Fig. 18 shows the distances between vehicle 12 and all the other vehicles throughout the manoeuvre. A vertical dashed line corresponds to the start of a new step in the formation change algorithm. The safety distance is set to 0.45 m. It can be seen that safety distance is kept between the vehicles. The other vehicles show similar behaviour. The minimum distance between two vehicles during the whole formation change process is 0.46 m.

5 Conclusion

The proposed path generation method together with a carefully tuned robust formation controller is capable of guaranteeing a safe formation change with a practically negligible constraint on the formation topology for any type of vehicles.

The developed method was applied to formation control of quadrotor helicopters in 4D (3D position and orientation Ψ).

The algorithm can be accelerated by performing computations in a distributed manner. Further methods with robust performance allowing constraints on the controller can also be taken into consideration, which are to be investigated in the near future.

References

1 **Book RV, Karp RM, Miller RE, Thatcher JW**, *Reducibility Among Combinatorial Problems.*, The Journal of Symbolic Logic, **40**(4), (Dec, 1975), 618+, DOI 10.2307/2271828.

2 **Fax J, Murray R**, *Information flow and cooperative control of vehicle formations*, IEEE Transactions on Automatic Control, **49**(9), (2004), 1465–1476, DOI 10.1109/TAC.2004.834433.

3 **Keviczky T, Borrelli F, Balas GJ**, *Decentralized receding horizon control for large scale dynamically decoupled systems*, Automatica, **42**(12), (2006), 2105 - 2115, DOI 10.1016/j.automatica.2006.07.008.

4 **Konc J, Janezic D**, *An improved branch and bound algorithm for the maximum clique problem*, MATCH Communications in Mathematical and in Computer Chemistry, (Jun, 2007), `http://www.sicmm.org/~konc/%C4%8CLANKI/MATCH58(3)569-590.pdf`.

5 **Massioni P**, *Decomposition Methods for Distributed Control and Identification*, Delft Center for Systems and Control, 2010.

6 **Massioni P, Keviczky T, Gill E, Verhaegen M**, *A Decomposition-Based Approach to Linear Time-Periodic Distributed Control of Satellite Formations*, Control Systems Technology, IEEE Transactions on, **19**(3), (May, 2011), 481 -492, DOI 10.1109/TCST.2010.2051228.

7 **Mayne DQ, Rawlings JB, Rao CV, Scokaert POM**, *Constrained model predictive control: Stability and optimality*, Automatica, **36**(6), (2000), 789 - 814, DOI 10.1016/S0005-1098(99)00214-9.

8 **Popov A, Werner H**, *A robust control approach to formation control*, Proc. European Control Conference, Budapest, Hungary, (2009), 4428–4433.

9 **Regula G, Lantos B**, *Backstepping based control design with state estimation and path tracking to an indoor quadrotor helicopter*, Periodica Polytechnica Electrical Engineering, **53**(3–4), (2009), 151–161, DOI 10.3311/pp.ee.2009-3-4.07.

10 **Rice JK, Verhaegen M**, *Distributed computations and control in multi-agent systems*, In: Proc. 4th Int. Conf. Autonomous Robots and Agents ICARA 2009, 2009, pp. 44–49, DOI 10.1109/ICARA.2000.4803923.

11 **Richards A, How J**, *Decentralized model predictive control of cooperating UAVs*, In: Decision and Control, 2004. CDC. 43rd IEEE Conference on, Vol. 4, 2004, pp. 4286 - 4291, DOI 10.1109/CDC.2004.1429425.

12 **Richards AG**, *Robust Constrained Model Predictive Control*, Massachusetts Institute of Technology, 2005.

13 **San Segundo P, Matia F, Rodriguez-Losada D, Hernando M**, *An improved bit parallel exact maximum clique algorithm*, Optimization Letters, (2011), 1–13, DOI 10.1007/S11590-011-0431-y.

14 **San Segundo P, Rodriguez-Losada D, Jiménez A**, *An exact bit-parallel algorithm for the maximum clique problem*, Computers & Operations Research, **38**(2), (2011), 571–581, DOI 10.1016/j.cor.2010.07.019.

15 **Schmidt MC, Samatova NF, Thomas K, Park BH**, *A scalable, parallel algorithm for maximal clique enumeration*, J. Parallel Distrib. Comput., **69**(4), (2009), 417–428, DOI 10.1016/j.jpdc.2009.01.003.

16 **Schouwenaars T, How J, Feron E**, *Decentralized Cooperative Trajectory Planning of Multiple Aircraft with Hard Safety Guarantees*, In: AIAA Guidance, Navigation, and Control Conference and Exhibit, 2004, p. pp. 14, DOI AIAA 2004-5141.

17 **Shaw E, Hedrick JK**, *String Stability Analysis for Heterogeneous Vehicle Strings*, In: American Control Conference, 2007. ACC '07, 2007, pp. 3118 -3125, DOI 10.1109/ACC.2007.4282789.

18 **Tomita E, Seki T**, *An efficient branch-and-bound algorithm for finding a maximum clique*, In: DMTCS'03: Proceedings of the 4th international conference on Discrete mathematics and theoretical computer science, Springer-Verlag; Berlin, Heidelberg, 2003, pp. 278–289, `http://portal.acm.org/citation.cfm?id=1783736`, DOI 10.1007/3-540-45066-1_22.

19 **Tomita E, Sutani Y, Higashi T, Takahashi S, Wakatsuki M**, *A Simple and Faster Branch-and-Bound Algorithm for Finding a Maximum Clique*, In: **Rahman MaF Satoshi** (ed.), WALCOM: Algorithms and Computation, Vol. 5942, Springer Berlin Heidelberg; Berlin, Heidelberg, 2010, pp. 191–203, `http://www.springerlink.com/content/7778800225080533`, DOI 10.1007/978-3-642-11440-3_18.

20 **Vassilevska V**, *Efficient algorithms for clique problems*, Information Processing Letters, **109**(4), (2009), 254–257, DOI 10.1016/j.ipl.2008.10.014.