Formation control driven by cooperative object tracking

Pedro U. Lima, Aamir Ahmad, André Dias, André G.S. Conceição, António Paulo Moreira, Eduardo Silva , Luis Almeida, Luis Oliveira, Tiago P. Nascimento

A B S T R A C T

In this paper we introduce a formation control loop that maximizes the performance of the cooperative perception of a tracked target by a team of mobile robots, while maintaining the team in formation, with a dynamically adjustable geometry which is a function of the quality of the target perception by the team. In the formation control loop, the controller module is a distributed non-linear model predictive controller and the estimator module fuses local estimates of the target state, obtained by a particle filter at each robot. The two modules and their integration are described in detail, including a real-time database associated to a wireless communication protocol that facilitates the exchange of state data while reducing collisions among team members. Simulation and real robot results for indoor and outdoor teams of different robots are presented. The results highlight how our method successfully enables a team of homogeneous robots to minimize the total uncertainty of the tracked target cooperative estimate while complying with performance criteria such as keeping a pre-set distance between the teammates and the target, avoiding collisions with teammates and/or surrounding obstacles.

Keywords:
Formation control
Formation state estimation
Model predictive control
Cooperative perception
Indoor soccer robots
Outdoor land and aerial robots
Target tracking

1. Introduction

Most of the past and current work on motion coordination of multiple (possibly heterogeneous) vehicles [1,2] focuses on controlling a vehicle formation with a given nominal geometry and a pre-determined trajectory or a static destination location, in some cases more [3] or less [4] compliant with the presence of obstacles on the formation trajectory. Such methods typically:

- assume full knowledge of the formation state, expressed as the relative distances and bearings among all the vehicles, and/or
- rely on local memory-less interactions, often jeopardizing global formation stability.

A vehicle formation is supposed to serve one or more mission objectives [5]. One such interesting case concerns localizing or tracking relevant objects, here and henceforth denominated as targets. Recent formation control methods go beyond simply stating the desired geometry for the formation by providing some meta-specifications (e.g., for velocity matching, connectivity maintenance and containment control among the formation members [6,7]) but often give little or no relevance to the requirements

imposed by target localization and/or tracking to the formation geometry, so as to improve the target detection and tracking quality (e.g., accuracy). Active cooperative perception methods in sensor and robot networks [8] concern precisely this problem: how to actively move mobile sensors so as to improve the accuracy of target detection by the network, as the result of (spatially and temporally) fusing the information from all the static and mobile sensors which observe the target during a step sequence. In this paper we propose an integrated solution of the "target localization and tracking by a vehicle formation" problem, supported on the following novel contributions:

- a cooperative target tracker based on a particle filter (PF) which estimates the target position and velocity;
- a non-linear model-predictive formation controller, with the control objective of efficiently tracking a target based on cooperative perception while achieving criteria such as minimizing the uncertainty about the target position, keeping a pre-set distance to the tracked object and/or avoiding collisions between teammates in the formation while tracking the target.

Therefore our solution integrates two basic modules: (i) controller and (ii) estimator.

Our controller consists of a distributed non-linear model predictive controller (DNMPC). Some predominant approaches in multi-robot formation control are: virtual structures, behavior-based and leader-following [9–11]. A widely used controller based on the leader-following approach is the model predictive controller (MPC) [12] which was recently introduced in holonomic robots [13]. The primary focus of most existing methods is only to maintain the formation based on pre-planned paths and static environment assumption. In a dynamically changing environment, if the trajectories are pre-defined, linear MPC applied to a non-linear system can still maintain a desired formation.

Recent approaches for active cooperative target tracking by a robot team formation such as [8] rely on computationally heavy optimization processes. By introducing the Gauss–Seidel relaxation in an iterative algorithm to detect the next best sensing location for the mobile sensors, the authors in [8] achieve a linearly growing computational complexity over methods like grid-based exhaustive search which have similar tracking accuracy but where the complexity grows exponentially with the number of sensors. The novelty in our approach of integrating the controller and estimator modules to achieve a formation that minimizes the joint uncertainty covariance of the tracked target lies in the fact that the controller module of each robot performs an optimization over an already fused target posterior which makes the computational complexity of the optimization process constant with respect to the number of mobile sensors in the team. Furthermore the decoupling of the optimization problem from the estimates fusion makes the approach more reliable in case of individual sensor or inter robot communication failures.

The field of cooperative target tracking has gained a lot of attention in the recent years [14–16]. Many efficient solutions such as decentralized PF for multiple target tracking [15] and global position sharing based on non-egocentric tracking of objects [16] have been proposed. These solutions focus more on compacting the data shared for communication bandwidth reduction and to overcome the problem of target occlusion. Some solutions such as [17] assume multiple static platforms and hence do not address the self-localization errors that creep in when using multiple mobile sensor platforms. The estimator in our work consists of a cooperative target tracker based on a PF described in full detail in previous work [18,19]. Essentially the core of it is a PF, modified to handle, within a single unified framework, the problem of complete or partial occlusion for some of the involved mobile sensor platforms, as well as inconsistent estimates in the global frame among sensors, due to observation errors and/or self-localization uncertainty of the sensor platforms. This acts as a feedback module providing the position and velocity estimates of the tracked object to the controller which in turn uses these estimates as well as the teammate positions to generate velocity set points for the robot running the integrated system.

The robots in our formations share information over wireless communication, which, given its low reliability, is another source of errors that increase cooperative perception noise. Beyond uncontrollable interferences inherent to the operational environment, typical wireless communication protocols are also subject to transmission collisions that lead to packet losses, which are particularly relevant when the robots share their states periodically in broadcast mode. Thus, we use a communication protocol that autosynchronizes the robot transmissions over the wireless medium, reducing collisions and improving the quality of the communication. We built upon the work in [20] to extend such protocol to ad-hoc networks that are better suited to robot teams [21]. We also used this protocol to provide an alternative relative localization system based on RF-ranging [21] later combined with signal strength information for faster localization assessment. The actual information sharing is carried out over a distributed shared memory middleware called Real-Time Data Base (RTDB) [22], which decouples local processing from communication delays and provides fast access to remote data with age information.

The rest of the article is organized as follows. The controller and the estimator modules are detailed in Section 2. Then we describe their integration in Section 3. This is followed by our approach's implementation details on our testbed and the experimental results in Section 4. We conclude with comments on future work in Section 5.

2. The controller and estimator modules

2.1. Controller module

The distributed non-linear model predictive controller (DNMPC-)based formation controller used in this work has its roots in the non-linear model predictive controller (NMPC) developed and implemented in one of our previous works [23]. NMPC has a partially distributed architecture where each robot calculates its own control inputs U solving its own optimization problem, and using a central unit only as a communication bridge. In the fully distributed architecture of the DNMPC the communication is performed by a real-time data base (RTDB) system [21]. This enables the robots to be communication-failure tolerant. Furthermore, even in the rare case of a communication failure, the robots use their predictive open-loop strategy to determine their teammate states making the DNMPC even more robust.

The DNMPC ability to create and maintain a formation is due to the fact that the cost functions used by the controllers of each robot in the team are coupled. This coupling occurs when the teammate states (position and velocity) are used in the cost function of each robot controller to enforce the desired formation geometry, thus the actions of each robot affect its teammates. The DNMPC iterates through the following two components:

- **optimizer**: uses an online numeric minimization method to optimize the cost function and generate the control signals. The resilient propagation (RPROP) method that is used here guarantees quick convergence;
- **predictor**: predicts the state evolution based on the system state model. The system consists of the robot itself, its teammates in the formation or another object in the environment with an impact on the formation objectives, such as a static obstacle or a moving target.

Therefore, the first step in designing the DNMPC-based formation controller is to establish the DNMPC cost function that enforces the desired formation geometry. Let there be N robots in a formation where the nth robot for $1 \leq n \leq \mathrm{N}$ is denoted by R_n. The DNMPC cost function $J(\mathrm{N_1}, \mathrm{N_2}, \mathrm{N_c})$ for robot R_n is given by (1):

$$\begin{aligned} J(\mathrm{N_1}, \mathrm{N_2}, \mathrm{N_c}) = & \sum_{i=\mathrm{N_1}}^{\mathrm{N_2}} \lambda_a |{}^i\mathcal{C}^{\perp}| + \sum_{i=\mathrm{N_1}}^{\mathrm{N_2}} \lambda_0 |(D_{\mathrm{val}} - \|{}^iP_{\mathrm{t}}^{\mathrm{R}_n}\|)| \\ & + \sum_{i=\mathrm{N_1}}^{\mathrm{N_2}} \lambda_1 |\delta(0, {}^i\theta_{\mathrm{t}}^{\mathrm{R}_n})| \\ & + \sum_{i=\mathrm{N_1}}^{\mathrm{N_2}} \lambda_2 |P_{\mathrm{val}} + ({}^i\tilde{P}_{\mathrm{t}}^{\mathrm{R}_n} \cdot {}^i\tilde{V}_t)| \\ & + \sum_{i=\mathrm{N_1}}^{\mathrm{N_2}} \sum_{j=1}^{\mathrm{N}} \lambda_3 \max\left(1 - \frac{\|{}^iP_{\mathrm{R}_n}^{\mathrm{R}_j}\|}{K_0}, 0\right) \\ & + \sum_{i=\mathrm{N_1}}^{\mathrm{N_2}} \sum_{k=0}^{\mathrm{N_b}} \lambda_4 \max\left(1 - \frac{\|{}^iP_{\mathrm{R}_n}^{\mathrm{O}_k}\|}{K_0}, 0\right) \\ & + \sum_{i=1}^{\mathrm{N_c}} \lambda_5 |\Delta U^i| \end{aligned} \tag{1}$$

where the left superscript assigned to any variable denotes its value at the ith iteration step of the DNMPC's optimizer and $|\cdot|$ denotes determinant for matrix arguments, 1-norm for vector arguments and absolute value for scalars. $\|\cdot\|$ represents the Euclidean norm. P_{t} and V_{t} are respectively defined as the position and velocity of the tracked target in the 2D global frame.

$$P_{\mathrm{t}} \overset{\text{def.}}{=} \begin{bmatrix} x_{\mathrm{t}} & y_{\mathrm{t}} \end{bmatrix}^{\top} \qquad V_{\mathrm{t}} \overset{\text{def.}}{=} \begin{bmatrix} v_{x\mathrm{t}} & v_{y\mathrm{t}} \end{bmatrix}^{\top} \tag{2}$$

where in the DNMPC the prediction module uses the following equations:

$$\begin{cases} x_{\mathrm{t}}(k) = x_{\mathrm{t}}(k-1) + T.(v_{x\mathrm{t}}(k)) \\ y_{\mathrm{t}}(k) = y_{\mathrm{t}}(k-1) + T.(v_{y_{\mathrm{t}}}(k)) \end{cases} \tag{3}$$

and

$$\begin{cases} v_{x\mathrm{t}}(k) = v_{x\mathrm{t}}(k-1).B_{\mathrm{FC}} \\ v_{y_{\mathrm{t}}}(k) = v_{y_{\mathrm{t}}}(k-1).B_{\mathrm{FC}} \end{cases} \tag{4}$$

with T as the time step and B_{FC} as the target friction coefficient.

$\tilde{V}_t$, defined as the target velocity unit vector, is given by:

$$\tilde{V}_t \overset{\text{def.}}{=} \begin{bmatrix} \tilde{v}_{x\mathrm{t}} & \tilde{v}_{y_{\mathrm{t}}} \end{bmatrix}^{\top} = \frac{V_{\mathrm{t}}}{\|V_{\mathrm{t}}\|}. \tag{5}$$

P_{R_n} and V_{R_n} are respectively defined as the pose and velocity of the robot R_n as follows:

$$\begin{aligned} P_{\mathrm{R}_n} &\overset{\text{def.}}{=} \begin{bmatrix} x_{\mathrm{R}_n} & y_{\mathrm{R}_n} & \theta_{\mathrm{R}_n} \end{bmatrix}^{\top} \\ V_{\mathrm{R}_n} &\overset{\text{def.}}{=} \begin{bmatrix} v_{x\mathrm{R}_n} & v_{y_{\mathrm{R}_n}} & w_{\mathrm{R}_n} \end{bmatrix}^{\top} \end{aligned} \tag{6}$$

and the dynamic of the robot state evolution is given by (assuming omnidirectional motion robots and a kinematic model only):

$$\begin{bmatrix} x_{\mathrm{R}_n}(k) \\ y_{\mathrm{R}_n}(k) \\ \theta_{\mathrm{R}_n}(k) \end{bmatrix} = \begin{bmatrix} x_{\mathrm{R}_n}(k-1) \\ y_{\mathrm{R}_n}(k-1) \\ \theta_{\mathrm{R}_n}(k-1) \end{bmatrix} + T. \begin{bmatrix} v_{x\mathrm{R}_n}(k) \\ v_{y\mathrm{R}_n}(k) \\ w_{\mathrm{R}_n}(k) \end{bmatrix}$$

and

$$\begin{bmatrix} v_{x\mathrm{R}_n}(k) \\ v_{y\mathrm{R}_n}(k) \\ w_{\mathrm{R}_n}(k) \end{bmatrix} = \begin{bmatrix} \cos(\theta_{\mathrm{R}_n}(k)) & -\sin(\theta_{\mathrm{R}_n}(k)) & 0 \\ \sin(\theta_{\mathrm{R}_n}(k)) & \cos(\theta_{\mathrm{R}_n}(k)) & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} v_{x_{\mathrm{ref}\mathrm{R}_n}}(k) \\ v_{y_{\mathrm{ref}\mathrm{R}_n}}(k) \\ w_{\mathrm{ref}\mathrm{R}_n}(k) \end{bmatrix}.$$

$P_{\mathrm{t}}^{\mathrm{R}_n}$ is defined as the position of the target in the robot R_n frame as follows:

$$P_{\mathrm{t}}^{\mathrm{R}_n} \overset{\text{def.}}{=} \begin{bmatrix} x_{\mathrm{t}}^{\mathrm{R}_n} & y_{\mathrm{t}}^{\mathrm{R}_n} \end{bmatrix}^{\top}, \tag{7}$$

where

$$\begin{cases} x_{\mathrm{t}}^{\mathrm{R}_n}(k) = x_{\mathrm{t}}(k) - x_{\mathrm{R}_n}(k) \\ y_{\mathrm{t}}^{\mathrm{R}_n}(k) = y_{\mathrm{t}}(k) - y_{\mathrm{R}_n}(k). \end{cases} \tag{8}$$

$\tilde{P}_{\mathrm{t}}^{\mathrm{R}_n}$, defined as the unit vector indicating the direction of the target w.r.t. the robot R_n, is given by:

$$\tilde{P}_{\mathrm{t}}^{\mathrm{R}_n} \overset{\text{def.}}{=} \begin{bmatrix} \tilde{x}_{\mathrm{t}}^{\mathrm{R}_n} & \tilde{y}_{\mathrm{t}}^{\mathrm{R}_n} \end{bmatrix}^{\top} = \frac{P_{\mathrm{t}}^{\mathrm{R}_n}}{\|P_{\mathrm{t}}^{\mathrm{R}_n}\|}. \tag{9}$$

$\theta_{\mathrm{t}}^{\mathrm{R}_n}$, defined as the bearing of the target w.r.t. the robot R_n is given by:

$$\theta_{\mathrm{t}}^{\mathrm{R}_n} = \operatorname{atan2}\left(\frac{y_{\mathrm{t}}^{\mathrm{R}_n}}{x_{\mathrm{t}}^{\mathrm{R}_n}}\right) \tag{10}$$

where atan2 has the usual meaning so as to calculate the appropriate quadrant of the computed angle.

$P_{\mathrm{R}_n}^{\mathrm{R}_j}$ is defined as the position of the robot R_n in robot R_j frame and $P_{\mathrm{R}_n}^{\mathrm{O}_k}$ is defined as R_n's position w.r.t. the kth obstacle for $1 \leq k \leq \mathrm{N_b}$ where $\mathrm{N_b}$ is the total number of obstacles in the environment. The poses of a robot R_n relative to its teammate R_j (where $1 \leq j \leq NM$, and NM is the total number of mates) are defined as:

$$P_{\mathrm{R}_n}^{\mathrm{R}_j}(k) = \begin{cases} x_{\mathrm{R}_n}^{\mathrm{R}_j}(k) = x_{\mathrm{R}_n}(k) - x_{\mathrm{R}_j}(k) \\ y_{\mathrm{R}_n}^{\mathrm{R}_j}(k) = y_{\mathrm{R}_n}(k) - y_{\mathrm{R}_j}(k) \end{cases} \tag{11}$$

and with respect to an obstacle O_l (where $1 \leq l \leq NO$, and NO is the total number of obstacles), is defined as:

$$P_{\mathrm{R}_n}^{\mathrm{O}_l}(k) = \begin{cases} x_{\mathrm{R}_n}^{\mathrm{O}_l}(k) = x_{\mathrm{R}_n}(k) - x_{\mathrm{O}_l}(k) \\ y_{\mathrm{R}_n}^{\mathrm{O}_l}(k) = y_{\mathrm{R}_n}(k) - y_{\mathrm{O}_l}(k). \end{cases} \tag{12}$$

It is important to mention that in the obstacle's state evolution, all obstacles (moving or static) are considered as having zero velocity at that time instant in order to reduce the computation load as the number of obstacles increases.

$C^{\perp}$ is the fused target position covariance matrix. D_{val} is the threshold distance between the robot and the target. P_{val} is the position coefficient which rotates the robots velocity towards a desired direction w.r.t. that of the target. ΔU^i is the control signal variation between two consecutive iterations of the optimizer. $\delta(\cdot)$ is defined as a function operating on two angles as arguments that returns their difference scaled between $-\pi$ and π. K_0 is the distance threshold between teammates or between a robot and an obstacle.

Finally, $\mathrm{N_1}, \mathrm{N_2}$ are the prediction horizon limits, in discrete time, such that $\mathrm{N_1} > 0$ and $\mathrm{N_2} \leq \mathrm{N_p}$ where $\mathrm{N_p}$ is the desired prediction horizon. $\mathrm{N_c}$ is the control horizon.

The cost function (1) involves seven terms, weighted by λ_a, $\lambda_0, \ldots, \lambda_5$, respectively. The formation geometry is enforced, in the next design step, by adequately setting the λ weights so as to balance the different constraints involved. The terms are described as follows:

- the first term (weighted by λ_a) concerns the penalization of the target (fused) uncertainty matrix determinant. This brings the robots into a geometric configuration (w.r.t. the target) that reduces the uncertainty of the target observation, according to the target measurement model of each robot sensor;

- the second term (weighted by λ_0) prevents the robots from colliding with the target;
- the third term (weighted by λ_1) keeps the robots' pose oriented towards the target position;
- the fourth term (weighted by λ_2) aligns the robots' velocity with the target velocity;
- the fifth and sixth terms (weighted by λ_3 and λ_4) prevent the inter-robot collisions and the robots' collisions with the static obstacles in the environment, respectively;
- the seventh and last term (weighted by λ_5) is the control effort penalization which restricts significant changes and oscillations in the control signal.

The third and last design step is the implementation of the controller algorithm. Iteration k of the algorithm goes as follows:

1. the optimizer, receives from the predictor the predicted state of the formation robots and of other relevant entities in the environment at step $k-1$, and calculates the control signal U that optimizes the cost function;
2. the predictor uses U and the system model to compute the formation state evolution for the following N_p prediction horizon steps.

These iterations continue until the minimum of the cost function is reached.

2.2. Estimator module

The estimator module is a cooperative target estimator (CTE) that provides to the DNMPC estimates of the target position (P_t) and velocity (V_t), together with the associated fused target position covariance matrix $C^{\perp}$.

The CTE consists of a PF with the standard prediction step but which, instead of the traditional update step [24], performs a fusion step as described in [19].

The first design step consists of determining the target measurement model (in this work case studies a Gaussian centered at the actual target position and with a covariance matrix that depends on the sensor used to measure the target position) and the target motion model (a constant velocity model with zero mean normally distributed acceleration noise [19]).

The second and last design step for the estimator module is the implementation of its algorithm. In our modified PF fusion step, a robot R_n transmits (i) its target observation measurement $z_t^{R_n}$ in the global frame; (ii) its target observation measurement confidence $C_t^{R_n}$; and (iii) its self-localization confidence C_{R_n}, to all its teammates and receives the same from them. $C_t^{R_n}$ is a value inversely proportional to the determinant of the target observation measurement covariance matrix. C_{R_n} is calculated depending on the algorithm used for the self-localization of the robots, e.g., in case Monte-Carlo localization (MCL) method is used, C_{R_n} is given by the effective number of particles in MCL [25,26]. Subsequently, the robot R_n builds[1] its own observation measurement pool (OMP) OMP_{R_n} which is a set defined as follows:

$$OMP_{R_n} = \{\langle z_t^{R_1}, \alpha^{R_1}\rangle, \ldots, \langle z_t^{R_N}, \alpha^{R_N}\rangle\} \tag{13}$$

where α^{R_i} for $i = 1, \ldots, N;\ i \neq n$ is the element weight (EW) assigned to the observation measurements in the pool received from teammate R_i and computed using its observation measurement and self-localization confidences, as follows:

$$\alpha^{R_i} = C_t^{R_i} C_{R_i}; \quad i = 1, \ldots, N;\ i \neq n. \tag{14}$$

[1] Please note that each robot in the team builds its own OMP on which its CTE operates.

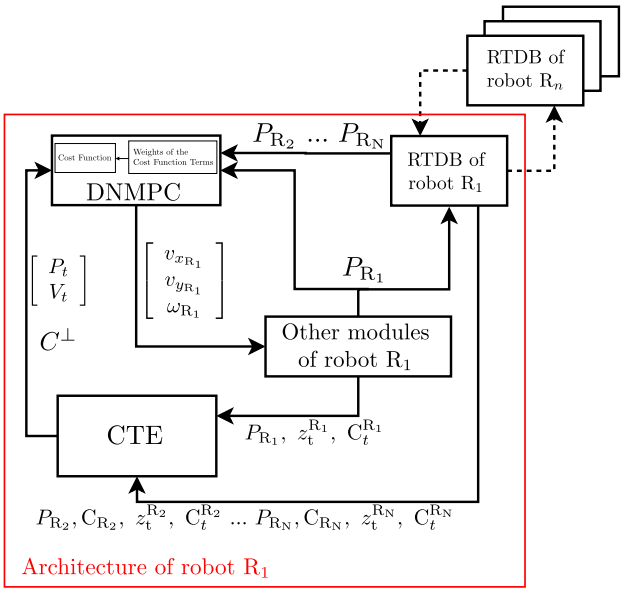

Fig. 1. Control–estimator module integration flow diagram for the robot R_1 (same for all the robots in the team). Variables in this figure are the ones defined in Section 2. The block named 'other modules of robot R_1' denotes that robot's low level control and sensor units, e.g., robot wheel controller and target detector (using camera images).

α^{R_n} is the EW assigned to the observation measurement in the pool made by the robot R_n itself and is computed using only R_n's observation measurement confidence:

$$\alpha^{R_i} = C_t^{R_i}; \quad i = n. \tag{15}$$

From here onwards the EWs are normalized.

For each particle, obtained from the PF's prediction step, an OMP element is drawn with a probability equal to its EW. The observation measurement corresponding to the drawn element is used to assign weight to a predicted particle. This is repeated for every particle in the predicted PF set. Finally, the PF resampling is performed in the usual way [24]. Through this process of confidence-based selection, each robot performs an update over its particle set in a way such that the information from all its teammates is accounted for without getting corrupted by their poor localization or noisy target observation measurements.

The CTE algorithm functions in a decentralized manner enabling each robot in the team to run its own instance of the CTE. Thus, like the DNMPC, it is resistant to individual robot's sensor failure and/or communication failure between any two robots, making it suitable for integration with the DNMPC.

3. Module integration

The central objective of the formation control in this work is to minimize the total uncertainty of the target cooperative estimate as perceived by the formation. This is achieved by integrating the controller and the estimator modules, through the incorporation in the DNMPC cost function of the fused target estimate obtained from the CTE, along with other terms concerning inter-robot distances, distance to the target, etc.

3.1. Functional integration

A flow diagram describing the integration of the control and the estimator modules is presented in Fig. 1. Each robot runs an instance of the DNMPC and the CTE. The CTE, represented as a single block in Fig. 1, communicates to the DNMPC the target

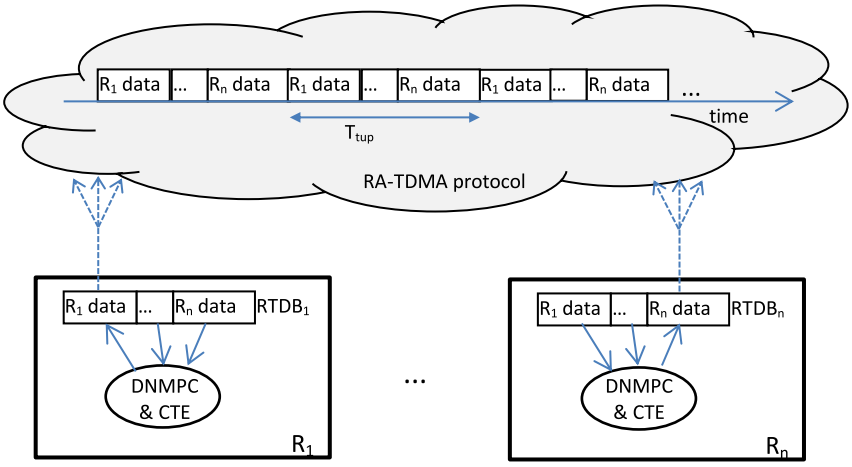

Fig. 2. Supporting integration with the RTDB.

position P_t, the target velocity V_t and the fused target position covariance matrix $C^{\perp}$. The DNMPC cost function at each robot also requires the teammates' positions. The cost function is formalized in Eq. (1).

The term involving $C^{\perp}$ in the cost function (1) is initialized with the fused target position covariance matrix received from the CTE. Subsequently, over the rest of the prediction horizons, the DNMPC predictor uses a pre-defined covariance evolution model, described further in Section 4, to predict the evolution of $C^{\perp}$ for the iterative minimization of the cost function. Once the minimum is reached, the DNMPC sends the velocity signals to the robot's wheel controllers so that the robot reaches the next best location maximizing the cooperative perception of the target while maintaining a pre-set threshold distance between teammates, between itself and the target as well as between itself and the obstacles in the environment.

3.2. Information integration

The actual integration of the control and estimator modules depicted in Fig. 2 is supported by a distributed shared memory middleware called RTDB [22]. This middleware provides, in each robot, proxies to remote data that are accessed locally similarly to local variables, thus without suffering communication delays. The proxies are updated in the background by a wireless communication protocol named RA-TDMA (Reconfigurable and Adaptive Time Division Multiple Access) [20].

For its internal computations, each robot requires the position of all teammates. Thus, each robot writes its own position in a shared variable in its RTDB interface and reads from this interface the position of its teammates Fig. 2. The RTDB middleware automatically captures the time when shared variables are written so that, when they are read, their age is computed and delivered to the application in a suitable structure. This allows detecting stale data and take appropriate action, such as discarding it or computing an estimate based on a suitable model.

The RA-TDMA protocol, on its side, transfers the contents of the shared variables from the robot where they are written to all others. This is carried out in a cyclic basis, called the team update period (T_{tup}), which is a configuration parameter. The transmissions are carried out in broadcast mode, which improves the temporal consistency across the RTDB, with all shared variables updated at once independently of the number of receivers.

This protocol has three distinctive features. On one hand, T_{tup} is divided in as many slots as robots that are currently active. An automatic reconfiguration takes places when a robot leaves the team, e.g., upon a crash, and when a new robot joins, e.g., reintegrating after a crash. On the other hand, these dynamic slots are enforced with an adaptive synchronization feature that keeps each robot transmitting in its slot, thus virtually eliminating collisions within the team. This synchronization is achieved based on the transmission instants of the other robots and does not require a globally synchronized clock.

Finally, the third feature is the operation in ad-hoc mode [21], without a central access point, in which the TDMA cycle reconfiguration and the slots synchronization are propagated through the network in an epidemic style.

4. Implementation, experiments and results

4.1. Indoor soccer robots

We implemented our formation controller on two separate RoboCup soccer middle size league (MSL) teams: (i) 5dpo [27] and (ii) SocRob [28]. The 5dpo robots (Fig. 3 (left)) consist of a 3-wheel omni-directional drive and a catadioptric vision system used for target state estimation. The SocRob robots, also based on a 3-wheeled omni-directional drive system, have a dioptric vision system containing a fish-eye lens camera facing downwards (Fig. 3 (right)). The tracked target was a standard FIFA size 5 soccer ball for both teams.

In this section we first present two separate covariance models for the evolution of the (fused) target position covariance matrix evolution, as explained in the previous section. These models were designed specifically for the robot team as well as the tracked object. Further implementation details, simulations and real robot results with an analysis are presented later.

4.1.1. 5dpo: target position covariance model

In the case of the 5dpo robots, we developed an empirical position covariance model $\mathcal{C}^{\perp}_{\text{5dpo}}$ (16) where the variance over the direction to the target is proportional to the squared distance to the target, and the variance over the perpendicular direction is proportional to the distance to the target.

$$\mathcal{C}^{\perp}_{\text{5dpo}} = \begin{bmatrix} K_1 d^2 & 0 \\ 0 & K_2 d \end{bmatrix}, \tag{16}$$

where K_1 and K_2 are constants of proportionality.

4.1.2. SocRob: target position covariance model

In the case of the SocRob robots, the position covariance model is derived from the dioptric system observation model expression.

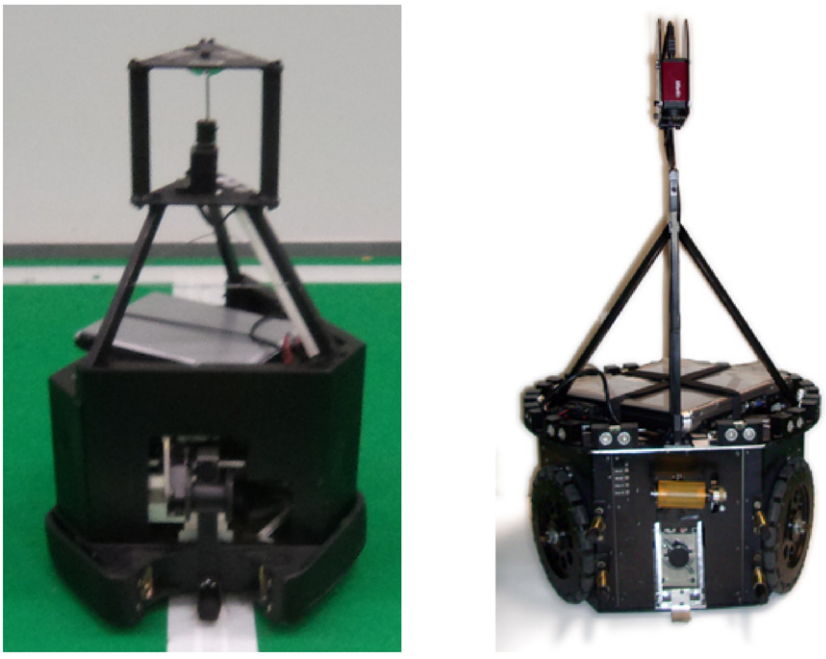

Fig. 3. 5dpo soccer robot with a catadioptric vision system (left). SocRob soccer robot with a dioptric vision system (right).

The robots have a fish-eye lens based dioptric vision system pointing downwards to the ground plane, the plane on which the robots roto-translate. The fish-eye lens' projection model is given by:

$$d_{\text{px}} = f\Theta, \tag{17}$$

where the 3D world frame follows a spherical coordinate system with coordinate variables denoted by r, Θ and Φ and the 2D image frame follows a polar coordinate system with coordinate variables d_{px} and Φ. Since Φ remains unchanged after the transformation, it is denoted by the same variable. f is the lens' field of view (FOV) constant. The origin of both the world frame and the image plane is assumed to be at $\mathcal{O}$, the point on the ground plane directly below the robot dioptric system's center.

The observation model expression for a spherical target of known size derived using (17) is given by (18) as

$$r_o^2 - r_o\sqrt{r_o^2 - R_o^2}\left(\sin\left(\frac{d_{\text{px}}}{f}\right)\sin\Theta_o\cos(\Phi - \Phi_o) + \cos\left(\frac{d_{\text{px}}}{f}\right)\cos\Theta_o\right) - R_o^2 = 0, \tag{18}$$

which is expressed in the image's polar coordinate variables d_{px} and Φ, the parameters r_o, Θ_o, Φ_o, the spherical coordinates of the target sphere's center and the fixed constants f (lens' FOV constant) and R_o (spherical target known radius).

Using the model (18), the target observation $\mathcal{M}_{\text{SocRob}}$ in these robots is a 3D vector $[r, \Theta, \Phi]$ for all $r_o \geq 0$, $0 \leq \Theta_o \leq \frac{\pi}{2}$, $-\pi \leq \Phi_o \leq \pi$. Assuming the estimates of each coordinate to be uncorrelated, the expression for the covariance model (19) is derived using (17) and (18).

$$\mathcal{C}_{\text{SocRob}} \overset{\text{def.}}{=} \begin{bmatrix} \sigma_r^2 & 0 & 0 \\ 0 & \sigma_\Theta^2 & 0 \\ 0 & 0 & \sigma_\Phi^2 \end{bmatrix} = \begin{bmatrix} K_3\left(\frac{r^2}{R_o^2}\right) & 0 & 0 \\ 0 & K_4\left(\frac{1}{r^2 - R_0^2}\right) & 0 \\ 0 & 0 & K_5\left(\frac{1}{r^2 - R_0^2\sin^2\Theta}\right) \end{bmatrix}. \tag{19}$$

The covariance model expression in (19) is first converted into the canonical form and then projected onto the ground plane (20) to simplify the process of covariance merging later.

$$\mathcal{C}_{\text{SocRob}}^{\perp} = \begin{bmatrix} \sigma_d^2 & 0 \\ 0 & \sigma_{d^\perp}^2 \end{bmatrix} \tag{20}$$

where

$$\sigma_d^2 = K_3\frac{r^2\sin^2\Theta}{2R_o^2} + K_4\frac{r^2\cos^2\Theta}{2(r^2 - R_0^2)} + K_3K_4\frac{r^2\cos^2\Theta}{4R_o^2(r^2 - R_0^2)},$$

$$\sigma_{d^\perp}^2 = K_5\frac{(r^2 + \sigma_d^2)}{2(r^2 - R_0^2\sin^2\Theta)}$$

and K_3, K_4 and K_5 are constants of proportionality.

4.1.3. Covariance merging

Covariance merging method using Smith and Cheeseman's formulation [29] is used to obtain the target position covariance as the result of fusing all robots' estimates, the evolution of which is simulated by the DNMPC predictor for the iterative minimization of the cost function. At each robot, $\mathcal{C}_{\text{5dpo}}^{\perp}$ in the case of 5dpo or $\mathcal{C}_{\text{SocRob}}^{\perp}$ in the case of SocRob robots, is predicted for the teammates in the formation, rotated in the robot's frame and then merged. In the case of simulation experiments, the means of observation estimates from teammates are identical while the uncertainty ellipse around each teammate's observation is formulated as per (16) or (20) for merging.

4.1.4. Implementation

The cost function (1) has several terms with changeable weights. In order to understand the behavior and influence of the covariance term that minimizes the determinant of the target fused uncertainty matrix, three situations were tested both in the simulations and in the real robots. The first situation penalizes the determinant of the uncertainty matrix and the control effort. The second penalizes the distances between the robots and the target, the collision between teammates and the control effort. The third situation assigns non-zero weights to every term in the cost function (1).

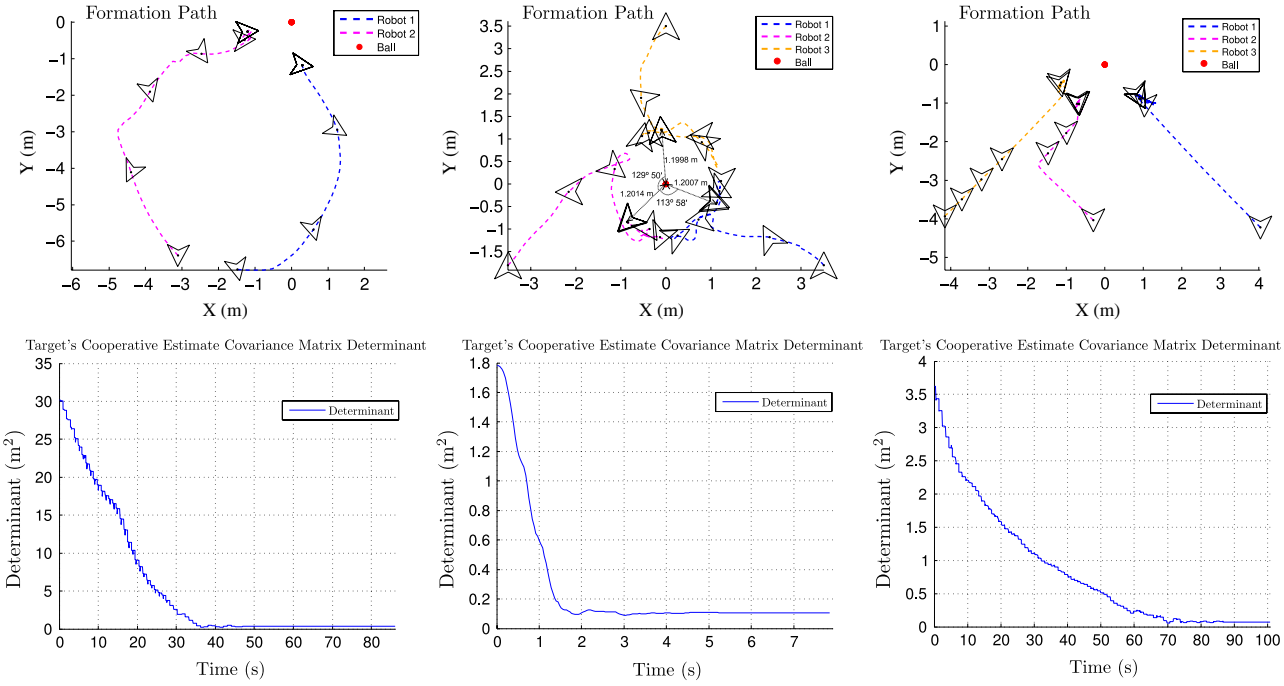

Fig. 4. Selection of simulated robot results: The plots in the first column are for SocRob's 2 robot case in the situation when only the target covariance term in the cost function (1) is active. The second column plots are for 5dpo's 3 robots case for the situation when all the terms in its cost function (1) are active. The third column is for SocRob's 3 robot case for the situation when only the target covariance term in the cost function (1) is active. The plots in the top row show the robots' trajectories in the formation and their final poses while achieving the cooperative minimum target uncertainty where as the plots in the bottom row, directly below the trajectories, show the corresponding evolution of the cooperative target estimates covariance determinant.

We present the results of the simulation and real robot experiments made with the 5dpo and SocRob robot soccer teams. The maximum allowed velocity in the 5dpo simulations was 1.4 m/s while in the SocRob simulations was 0.5 m/s due to the differences in the dynamics of the robots. Each team uses its own target position covariance model as described earlier in this section. The 5dpo simulations were made using SimTwo[2] and the SocRob's were made using the Webots simulator.[3]

For each team, the first set of simulation experiments is comprised of 2 robots while the second set is comprised of 3 robots. Results of all these simulations are presented. Robots' trajectory plots and target fused position covariance determinant plots of selected experiments are also presented (Fig. 4). Real robot experiments were performed using a team of 2 SocRob robots. The video attached with this paper shows the simulation and real robot experiments' footage where the trajectory for each robot and the formation can be visualized (see video at http://youtu.be/lMA8gHa6iZI).

4.1.5. Simulation results

Tables 1 and 2 show the final value of the determinant of the target fused position covariance matrix after minimization. Table 3 summarizes the weights used for each term of the cost function (1) during the experiments. It should be noted that the SocRob's target position covariance model is derived from its observation model itself and hence it naturally finds the best position w.r.t. the target for minimizing the fused target position covariance whereas the 5dpo's model is built empirically for which the minimization occurs when the robot's and target positions coincide. Therefore in the cost function (1) the robot–target distance threshold term's weight λ_0 for the 'only target covariance' experiment situation in case of 5dpo robot was set to a positive value to avoid robot–target

Table 1
Determinant of the target fused position covariance matrix, obtained from simulations with 2 5dpo robots and 2 SocRob robots, for three combinations of the DNMPC cost function terms, dubbed as 'Situations' here and detailed in Table 3.

5dpo—2 robots case	
Situation	$\lvert\mathcal{C}^{\perp}_{\text{5dpo}}\rvert$
Only target covariance	0.2252
Only mates	0.3145
All terms	0.2201
SocRob—2 robots case	
Situation	$\lvert\mathcal{C}^{\perp}_{\text{SocRob}}\rvert$
Only target covariance	0.3356
Only mates	0.4205
All terms	0.3415

Table 2
Determinant of the target fused position covariance matrix, obtained from simulations with 3 5dpo robots and 3 SocRob robots, for three combinations of the DNMPC cost function terms, dubbed as 'Situations' here and detailed in Table 3.

5dpo—3 robots case	
Situation	$\lvert\mathcal{C}^{\perp}_{\text{5dpo}}\rvert$
Only target covariance	0.1017
Only mates	0.1095
All terms	0.1074
SocRob—3 robots case	
Situation	$\lvert\mathcal{C}^{\perp}_{\text{SocRob}}\rvert$
Only target covariance	0.0742
Only mates	0.0646
All terms	0.0924

[2] Paco Wiki—SimTwo http://paginas.fe.up.pt/~paco/wiki/index.php?n=Main.SimTwo.

[3] Cyberbotics Ltd.—Webots, http://www.cyberbotics.com/overview.

Table 3
Values of weights for all simulation experiments. The weights were the same for both the 2 and 3-robots cases.

	Only target covariance		Only mates		All terms	
	5dpo	*SocRob*	*5dpo*	*SocRob*	*5dpo*	*SocRob*
λ_a	3000	300	0	0	3000	300
λ_0	1500	0	1500	1500	1500	1500
λ_1	0	0	0	0	300	200
λ_2	0	0	0	0	100	100
λ_3	0	0	500	600	500	600
λ_4	0	0	0	0	600	600
λ_5	5	100	5	100	5	100

Table 4
Determinant of the target fused position covariance matrix, obtained from two SocRob real robots, for three combinations of the DNMPC cost function terms, dubbed as 'Situations' here and detailed in Table 3.

Situation	$\lvert\mathcal{C}^{\perp}_{\text{SocRob}}\rvert$
Only target covariance	0.687
Only mates	0.703
All terms	0.233

Fig. 5. TIGRE.

Fig. 6. Asctec@ Pelican MAV.

collision whereas λ_0 for the same situation in SocRob's need not be positive.

4.1.6. Real robot results

The video attached with this paper presents the footage of the 2 SocRob real robot's experiments. The weights for each term in (1) were the same as in the simulations. Table 4 shows the final determinant value of the target fused position covariance matrix after minimization. It should be noted that for the real robot's case when all the terms of the cost function are active, a lower determinant value of the target fused position covariance matrix is obtained. This is because when only the target position covariance term is used in the cost function the chances for the formation to get stuck in a local minimum of the DNMPC cost function are higher. Hence, introducing the other terms in the cost function, such as teammate avoidance and orientation towards the target, helps the formation converging to a global minima.

Fig. 7. Experimental scenario.

Fig. 8. Dynamic target tracking by TIGER and Pelican.

4.2. Outdoor field robots

In the outdoor scenario, we implemented our formation CTE on a team of 2 robots: (i) TIGRE; and (ii) Pelican. The robot TIGRE [30] (see Fig. 5) is an autonomous ground robot for exploration and activity in unstructured environments. It has an electric propulsion engine and is equipped with an on-board Quad Core Intel(R) Core(TM) i5 CPU 750 @ 2.67 GHz processor with 4 GB RAM, a wireless network card, an infra-red thermographic camera, a laser rangefinder, two visible spectrum cameras in a rigid stereo baseline (∼0.76 m) with a pixel resolution of 1278 × 958, a Novatel GPS receiver and a Microstrain IMU.

The robot Pelican, a Micro Aerial Vehicle (MAV) (see Fig. 6), is a helicopter driven by four rotors symmetric to the center of mass. It is equipped with a Flight Control Unit (FCU) for data fusion (from a GPS and an IMU), an on-board 1.6 GHz Intel Atom-based embedded computer, an 802.11n Wifi and a monocular camera from IDS UEYe LE with a resolution of 1280 × 1024.

Both robots run Linux and the ROS framework as a middleware for parametrization and monitoring of all processes. In addition, RTDB and RA-TDMA are used to share data between robots.

4.2.1. Experimental scenario and implementation

The experiments with the outdoor robots were performed in a non-urban area with several landscape elements, e.g., vegetation, water, rocks, bushes and some semi-urban structures such as gravel paths (see Fig. 7). Both TIGRE and Pelican were tele-operated in this scenario. A person wearing a 37 cm × 67 cm orange life jacket (see Fig. 8) walked in the environment to emulate

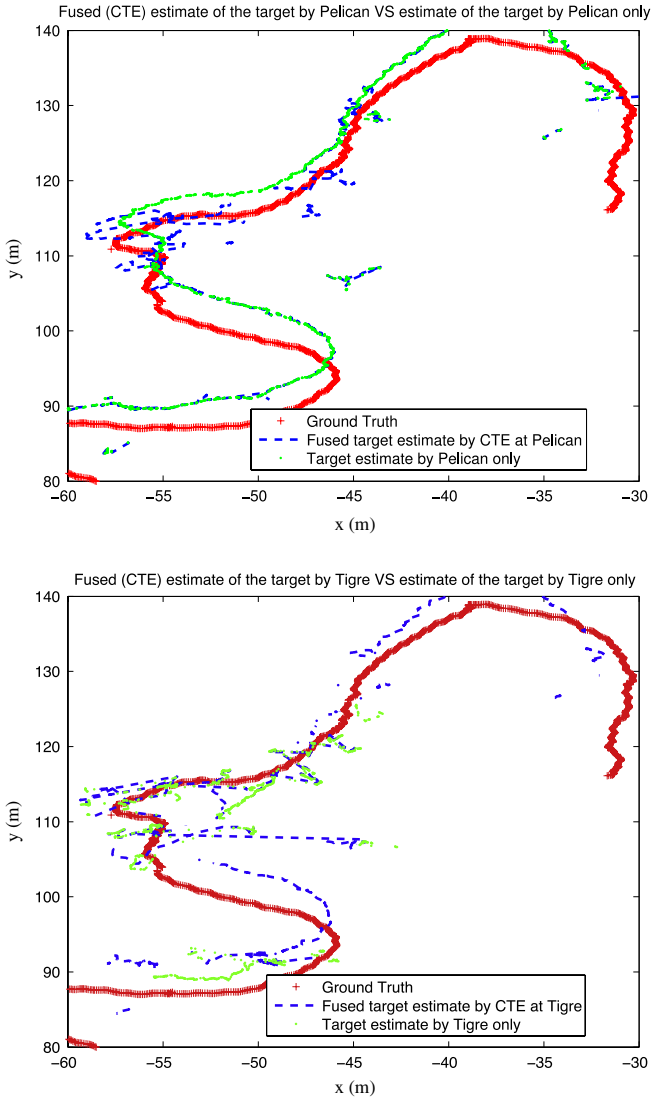

Fig. 9. Trajectory plots of the tracked target's position estimates by the individual robots and by the CTE running on each of them compared against the ground truth.

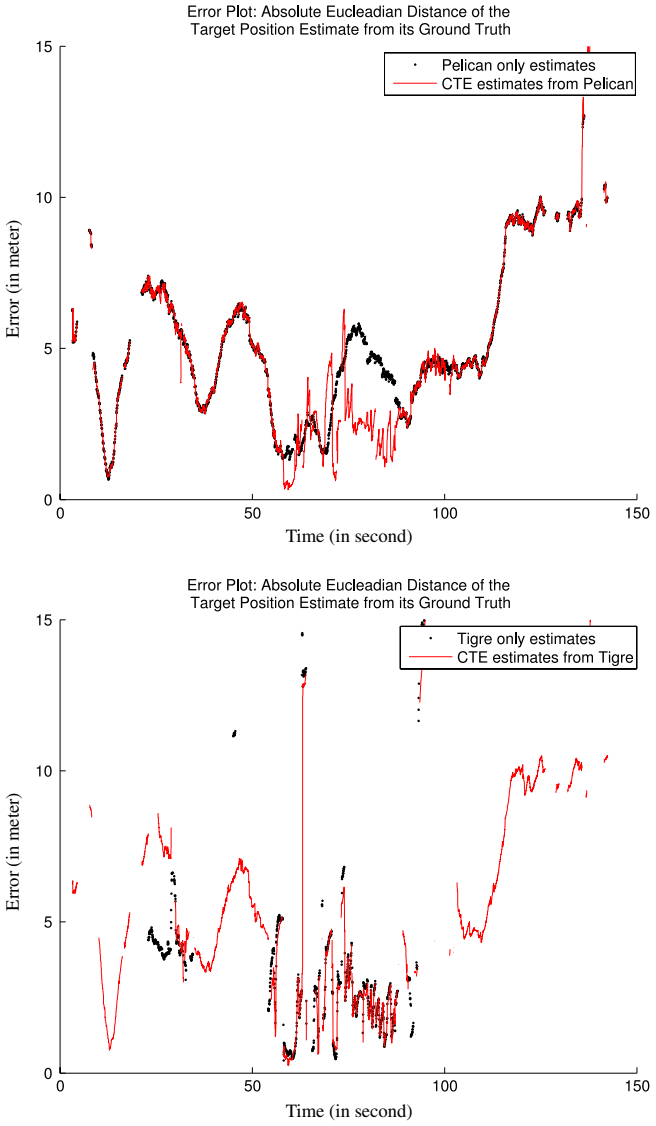

Fig. 10. Error plots of the tracked target's position estimates w.r.t. the ground truth.

the moving target. The life jacket was equipped with an RTK GPS Septentrio L1 L2 able to provide its GPS coordinates (during post-processing) with the maximum error of 10 cm. The position estimates from this GPS were considered to be the ground truth positions of the target and were used to evaluate the accuracy of our proposed method's implementation.

Images acquired by the cameras mounted on both the robots and their GPS localization were logged in the respective robot's computers along with the associated timestamps. These images were post-processed to obtain the target observation measurements. The CTE was then applied on these measurements (robot localization and target position data). Note that Pelican possesses only one camera and it does not support 3D target detection at this stage. Consequently, for these preliminary results we considered the target's ground height to be constant. In Section 4.2.2 we present the preliminary results of these experiments that include the cooperatively estimated positions of the target compared with its ground truth positions. Please note that currently the formation controller DNMPC is not fully implemented on the outdoor robots, hence unlike the indoor robots experimental results (described previously in Section 4.1) the complete perception-driven formation control experiments for the outdoor robots case are not presented. Including the DNMPC in the control closed loop is currently ongoing work.

4.2.2. *Real robots preliminary results*

Fig. 9 presents the trajectories of the target as estimated by each of the robots individually and by the CTE running on these robots. Fig. 10 presents the corresponding error plots of these estimates. The error is calculated as the absolute value of the Euclidean distance between the estimated positions of the target and its corresponding ground truth positions. The statistical estimates of these errors are presented in Table 5. Although the mean/median tracking error for both robots seems large (4–5 m), it must be noted that these experiments were performed over an area of $\sim 10^4$ m^2, and that the average GPS localization errors of the robots are $\sim$1 m. A clear benefit of using the CTE over the individual robot's estimate is the increase in the tracking period achieved by both robots. For large time periods during the experiment (where the total experiment time corresponds to $\sim$150 s), the target moved out of the field of view of one of the robots or of both. However, Pelican could track the target for 31.43% of the time more by using the CTE instead of simply using its own observation measurements

Table 5
Error statistics of the tracked position estimates of the target. P.I.T.P. stands for percentage increase in tracking period. It refers to the time period in which a robot was not directly observing the target but was able to track it due to the cooperative tracking by the CTE running on it.

Robot	Individual estimate			CTE estimates			P.I.T.P.
	Mean (m)	Median (m)	Std. Dev. (m)	Mean (m)	Median (m)	Std. Dev. (m)	
Pelican	5.71	4.62	4.56	5.65	4.48	5.05	31.43%
TIGRE	6.10	3.55	6.73	6.83	4.96	5.94	51.29%

for tracking, whereas the increase in TIGRE's tracking period was 51.29% when using the CTE.

5. Conclusions and future work

A method for multi-robot formation control with the main goal of tracking a target with minimum uncertainty was proposed in this paper. The control loop includes a distributed nonlinear model predictive controller (DNMPC) and a cooperative target state (position and velocity) estimator (CTE). The DNMPC controls the formation geometry dynamically to minimize a cost functional consisting of several terms, one of them being the fused target position covariance matrix determined by the CTE, while the CTE uses a particle filter to fuse the vision-based observation measurements of the target by all the formation robots. The inter-robot wireless communications are efficiently handled based on a wireless communication protocol (RA-TDMA) and a real-rime database (RTDB).

Simulation and real robot results were presented for indoor and outdoor heterogeneous robots demonstrating the comprehensiveness of the method implementation. CTE is capable of handling different situations of occlusion, observation noise and camera models to fuse their information and determine the target state accurately, while providing the associated estimate uncertainty. The DNMPC cost functional enables balancing different formation control objectives, such as obstacle avoidance, collision avoidance, or improving target estimation accuracy, while creating and maintaining the robot formation.

Our future work includes extending this approach to heterogeneous robot teams composed of both holonomic and non-holonomic robots, as well as to implementing it fully in outdoor robots.

In this paper we have focused on the experimental results of the proposed technique combining a cooperative target tracker based on a particle filter and a distributed non-linear model-predictive formation controller. Formation control stability analysis and other related theoretical issues [31,32] should be tackled in future work. Nevertheless, one should note that the most accepted technique used to prove stability in MPC with constraints relies on the use of a positive invariant set as a terminal constraint. There is a number of methods for determination of such a set in the linear case. In the nonlinear case, there is no systematic way of doing that, to our knowledge. The use of a cascade structure with nonlinear elements makes this task even more difficult. Regarding the estimator, particle filters are one of the most versatile Bayesian filters that have been extensively used in many robotic applications. As particle filters do not make strong parametric assumptions on the posterior belief densities, they are particularly suited for complex multimodal distributions. It is well known that theoretically the approximation error in a PF converges to zero only when the number of particles approximating the posterior density tends to infinity. In practice, however, resource-adaptive number of particles produce sufficiently accurate approximations of the posterior if the observations are handled in a smart manner. In our work, the measurements are multimodal and arise from various sources (teammates observing the same object) where each source in itself is uncertain of its own location. By weighting the measurements from these sources proportionally to the sources' own localization confidence as well as their measurement confidence of the target, we do not only mitigate the error in target's estimate in the world frame but also prevent a systematic bias in the target estimate that might occur in case only a single measurement source existed and number of particles were not very high.

Acknowledgments

This work was funded by FCT — Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) under project FCT PTDC/EEA-CRO/100692/2008. The work of A. Dias and E. Silva was also funded by ERDF — European Regional Development Fund through the COMPETE Programme (operational programme for competitiveness) and by FCT under project FCOMP-01-0124-FEDER-037281.

References

[1] T. Keviczky, F. Borrelli, K. Fregene, D. Godbole, G.J. Balas, Decentralized receding horizon control and coordination of autonomous vehicle formations, IEEE Trans. Control Syst. Technol. 16 (1) (2008) 19–33.
[2] R.M. Murray, Recent research in cooperative control of multi-vehicle systems, J. Dyn. Syst. Meas. Control-Trans. ASME 129 (5) (2007) 571–583.
[3] P. Ogren, E. Fiorelli, N.E. Leonard, Cooperative control of mobile sensor networks: Adaptive gradient climbing in a distributed environment, IEEE Trans. Automat. Control 49 (8) (2003) 1292–1302.
[4] J.P. Desai, J.P. Ostrowski, V. Kumar, Modeling and control of formations of nonholonomic mobile robots, IEEE Trans. Robot. Autom. 17 (6) (1999).
[5] F. Zhang, N. Leonard, Cooperative filters and control for cooperative exploration, IEEE Trans. Automat. Control 55 (3) (2010) 650–663. http://dx.doi.org/10.1109/TAC.2009.2039240.
[6] Z. Meng, Z. Lin, W. Ren, Leader–follower swarm tracking for networked Lagrange systems, Systems Control Lett. 61 (1) (2012) 117–126.
[7] Y. Cao, D. Stuart, W. Ren, Z. Meng, Distributed containment control for multiple autonomous vehicles with double-integrator dynamics: algorithms and experiments, IEEE Trans. Control Syst. Technol. 19 (4) (2011) 929–938.
[8] K. Zhou, S. Roumeliotis, Multirobot active target tracking with combinations of relative observations, IEEE Trans. Robot. 27 (4) (2011) 678–695. http://dx.doi.org/10.1109/TRO.2011.2114734.
[9] M. Daigle, X. Koutsoukos, G. Biswas, Distributed diagnosis in formations of mobile robots, IEEE Trans. Robot. 23 (2) (2007) 353–369. http://dx.doi.org/10.1109/TRO.2007.895081.
[10] D. Gu, A differential game approach to formation control, IEEE Trans. Control Syst. Technol. 16 (1) (2008) 85–93. http://dx.doi.org/10.1109/TCST.2007.899732.
[11] Y. Ding, Y. He, Flexible leadership in obstacle environment, in: 2010 International Conference on Intelligent Control and Information Processing, ICICIP, 2010, pp. 788–791. http://dx.doi.org/10.1109/ICICIP.2010.5564314.
[12] W. Dunbar, R. Murray, Model predictive control of coordinated multi-vehicle formations, in: Decision and Control, 2002, Proceedings of the 41st IEEE Conference on, Vol. 4, 2002, pp. 4631–4636. http://dx.doi.org/10.1109/CDC.2002.1185108.
[13] K. Kanjanawanishkul, A. Zell, A model-predictive approach to formation control of omnidirectional mobile robots, in: Intelligent Robots and Systems, 2008. IROS 2008, IEEE/RSJ International Conference on, 2008, pp. 2771–2776. http://dx.doi.org/10.1109/IROS.2008.4650752.
[14] L.-L. Ong, B. Upcroft, T. Bailey, M. Ridley, S. Sukkarieh, H. Durrant-Whyte, A decentralised particle filtering algorithm for multi-target tracking across multiple flight vehicles, in: Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on, 2006, pp. 4539–4544. http://dx.doi.org/10.1109/IROS.2006.282155.

[15] L.-L. Ong, B. Upcroft, M. Ridley, T. Bailey, S. Sukkarieh, H. Durrant-Whyte, Consistent methods for decentralised data fusion using particle filters, in: Multisensor Fusion and Integration for Intelligent Systems, 2006 IEEE International Conference on, 2006, pp. 85–91. http://dx.doi.org/10.1109/MFI.2006.265604.

[16] D. Gohring, H.-D. Burkhard, Multi robot object tracking and self localization using visual percept relations, in: Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on, 2006, pp. 31–36. http://dx.doi.org/10.1109/IROS.2006.282427.

[17] X. Sheng, Y.-H. Hu, P. Ramanathan, Distributed particle filter with gmm approximation for multiple targets localization and tracking in wireless sensor network, in: Information Processing in Sensor Networks, 2005. IPSN 2005, Fourth International Symposium on, 2005, pp. 181–188. http://dx.doi.org/10.1109/IPSN.2005.1440923.

[18] A. Ahmad, P.U. Lima, Multi-robot cooperative object tracking based on particle filters, in: Proc. of the European Conference on Mobile Robots, ECMR 2011, Örebro, Sweden, 2011.

[19] A. Ahmad, P.U. Lima, Multi-robot cooperative spherical-object tracking in 3D space based on particle filters, Robot. Auton. Syst. 61 (10) (2013) 1084–1093.

[20] F. Santos, L. Almeida, L. Lopes, Self-configuration of an adaptive TDMA wireless communication protocol for teams of mobile robots, in: Emerging Technologies and Factory Automation, 2008. ETFA 2008, IEEE International Conference on, 2008, pp. 1197–1204. http://dx.doi.org/10.1109/ETFA.2008.4638554.

[21] L. Oliveira, L. Almeida, F. Santos, A loose synchronisation protocol for managing RF ranging in mobile ad-hoc networks, in: T. Röfer, N. Mayer, J. Savage, U. Saranli (Eds.), RoboCup 2011: Robot Soccer World Cup XV, in: Lecture Notes in Computer Science, vol. 7416, Springer, Berlin, Heidelberg, 2012, pp. 574–585.

[22] F. Santos, L. Almeida, P. Pedreiras, L. Lopes, A real-time distributed software infrastructure for cooperating mobile autonomous robots, in: Advanced Robotics, 2009. ICAR 2009, International Conference on, 2009, pp. 1–6.

[23] T.P. Nascimento, F.A. Fontes, A.P. Moreira, A.G.S. Conceição, Leader following formation control for omnidirectional mobile robots—the target chasing problem, in: J.-L. Ferrier, A. Bernard, O.Y. Gusikhin, K. Madani (Eds.), ICINCO (2), SciTePress, 2011, pp. 135–144.

[24] S. Thrun, W. Burgard, D. Fox, Probabilistic Robotics (Intelligent Robotics and Autonomous Agents), The MIT Press, 2005.

[25] J.S. Liu, Metropolized independent sampling with comparisons to rejection sampling and importance sampling, Stat. Comput. 6 (1996) 113–119. http://dx.doi.org/10.1007/BF00162521.

[26] G. Grisettiyz, C. Stachniss, W. Burgard, Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling, in: Robotics and Automation, 2005. ICRA 2005, Proceedings of the 2005 IEEE International Conference on, 2005, pp. 2432–2437. http://dx.doi.org/10.1109/ROBOT.2005.1570477.

[27] T. Nascimento, M.A. Pinto, H.M. Sobreira, F. Guedes, P. Castro, A. Malheiros, A. Pinto, H.P. Alves, M. Ferreira, P. Costa, P.G. Costa, A. Souza, L. Almeida, L.P. Reis, A.P. Moreira, 5dpo team description paper, in: RoboCup 2011, Istanbul, Turkey.

[28] J. Messias, A. Ahmad, J. Reis, J. Sousa, P.U. Lima, Socrob team description paper, in: RoboCup 2011, Istanbul, Turkey.

[29] R.C. Smith, P. Cheeseman, On the representation and estimation of spatial uncertainly, Int. J. Robot. Res. 5 (4) (1986) 56–68. http://dx.doi.org/10.1177/027836498600500404.

[30] A. Martins, G. Silva, A. Dias, C. Almeida, J. Almeida, E. Silva, TIGRE—an autonomous ground robot for outdoor exploration, in: Robotica 2013: 13th Conference on Autonomous Robot Systems and Competitions, 2013.

[31] Y. Chen, Z. Wang, Formation control: a review and a new consideration, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2005, 2005, pp. 3181–3186.

[32] R. Olfati-Saber, Flocking for multi-agent dynamic systems: Algorithms and theory, IEEE Trans. Automat. Control 51 (3) (2006) 401–420.