Dynamic Placement with Connectivity for RSNs based on a Primal-Dual Neural Network

Rafael Lima de Carvalho, Lunlong Zhong, Felipe Maia Galvão França, Felix Mora-Camino

Dynamic Placement with Connectivity for RSNs based on a Primal-Dual Neural Network

Rafael L. Carvalho[1,2] Lunlong Zhong[1] Felipe M. G. França[2] and Felix Mora-Camino[1]

[1] ENAC, MAIAA, Univ. de Toulouse, F-31055 Toulouse, France
[2] PESC, COPPE, Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil

Abstract. The present work deals with the dynamic placement of a set of pursuers and a set of relay devices so that the mean distance to a set of moving targets is minimized along a given period of time. The relay devices are here in charge of maintaining the communication between the pursuers. Moving targets, relay devices and pursuers are limited in their movements from one period to the next. The periodic problem is formulated as a linear quadratic programming model and a primal-dual neural network is proposed to solve from one stage to the next the current optimization problem. Moreover, the feasibility of the proposed approach is displayed through a numerical example.

1 Introduction

A Robotic Sensor Network (RSN) is a network composed of a set of entities with mobility and sensing capabilities. Here a set of pursuers and a set of relay devices must be located from one period to the next so that the mean distance to a set of moving targets is minimized along a given period of time.

As RSNs allow the possibility of harder missions, they also inherit the problems of related areas (sensor networks and mobile robots). In [1], the authors have proposed a distributed control framework, based on potential fields, that simultaneously addresses the velocity setting and connectivity requirement in the alignment problem of mobile agents. Other authors [2] have developed an adaptive algorithm based on local gradients of the signal-to-noise ratio in the communication links to control the motion of 2D relays vehicles. The control of a chain of robots between a reference robot and an explorer one, where at each step the relay robots move in such a way as to improve the Fielder value of weighted Laplacian, has been considered in [3]. The Fielder value describes in that case the communication interactions of all robots. Moreover, the authors of [4] considered the problem of dimensioning a set of robotic routers to provide connection between a user robot and a base station within an environment with obstacles.

In this paper we consider the problem of positioning pursuers and relays in a way that minimizes the mean distance between targets and pursuers, subject to movement and connectivity constraints. Our approach consists in formulating at each stage a linear quadratic problem. This problem is solved by a recurrent neural network associated to its primal-dual optimality conditions. After introducing the considered problem, the remainder of the paper is divided into four sections. The Section 2 presents the quadratic formulation of the problem. Afterwards, the primal-dual neural network solver is discussed in Section 3. Further, numerical results are displayed and

analyzed in Section 4. Finally, Section 5 concludes this report with some final remarks and future directions for research in this field.

2 Tracking targets with connectivity constraints

In this study we consider the case in which a set of N agents are pursuing a set of N targets on a limited Cartesian plane such as:

$$X_{min} \leq x \leq X_{max} \quad (1.1)$$
$$Y_{min} \leq y \leq Y_{max} \quad (1.2)$$

while a set of M mobile connecting devices (relays) are used to maintain communication between the pursuers. It is supposed that to each pursuer p_i is assigned a target t_i, i=1 to M, while the communication structure is given by a tree whose nodes are relays c_j, j=1 to M and leaves are pursuers as shown in Figure 1.

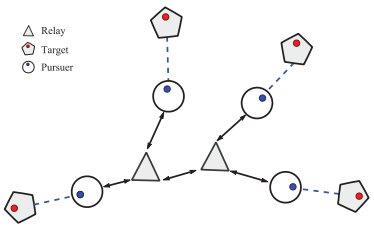

Fig. 1 A scenario with 4 targets, 4 pursuers and 2 relay nodes.

Here, time is discretized and at time k-1 the positions of the targets are given by their Cartesian coordinates $X_i^{k-1}, Y_i^{k-1}, i = 1, \dots, N$ while the positions of the pursuers are given by their Cartesian coordinates $x_i^{k-1}, y_i^{k-1}, i = 1, \dots, N$ as well as the position of the connecting nodes $x_i^{k-1}, y_i^{k-1}, i = N+1, \dots, N+M$.

Consider that the connecting tree T composed of N+M vertices is given by an arborescence starting at vertex p_1, so that the tree can be described by the successors of each vertex through the descendant function Γ_T. Here it is considered that connectivity is achieved at time $k-1$ when:

$$if\ j \in \Gamma_T^i, i = 1, \dots, N+M:$$
$$-d \leq x_i^{k-1} - x_j^{k-1} \leq d \quad (2.1)$$
$$-d \leq y_i^{k-1} - y_j^{k-1} \leq d \quad (2.2)$$

while

$$x_i^{k-1} \in [X_{min}, X_{max}], y_i^{k-1} \in [Y_{min}, Y_{max}], for\ i = 1, \dots, N+M \quad (3)$$
$$X_i^{k-1} \in [X_{min}, X_{max}], Y_i^{k-1} \in [Y_{min}, Y_{max}], for\ i = 1, \dots, N \quad (4)$$

It is also supposed that from a time period to the next, the position change of the targets is limited in a way such as for i=1 to N:

$$-\delta \le X_i^{k-1} - X_i^k \le \delta \tag{5.1}$$
$$-\delta \le Y_i^{k-1} - Y_i^k \le \delta \tag{5.2}$$

where $\delta < d$ while

$$X_i^k \in [X_{min}, X_{max}],\ Y_i^k \in [Y_{min}, Y_{max}]\ , for\ i = 1, \dots, N \tag{6}$$

Now we consider that the pursuers take positions at stage k such as they minimize the performance index given by:

$$\bar{d}_k = \frac{1}{N}\sqrt{\sum_{i=1}^{N}\left((x_i^k - X_i^k)^2 + (y_i^k - Y_i^k)^2\right)} \tag{7}$$

which is the mean distance between pursuers and targets, while satisfying the above constraints (2) to (6) and a set of step size limitations constraints expressed as:

$$-r + x_i^{k-1} \le x_i^k \le r + x_i^{k-1} \qquad i = 1, \cdots, N \tag{8.1}$$
$$-r + x_i^{k-1} \le x_i^k \le r + x_i^{k-1} \qquad i = 1, \cdots, N \tag{8.1}$$

Then the solution of this problem at stage k is identical to the solution of the linear quadratic mathematical programming problem given by:

$$\min_{x_i^k, y_i^k, i=1,\dots,N} \sum_{i=1}^{i=N}\left(\left(x_i^k - X_i^k\right)^2 + \left(y_i^k - Y_i^k\right)^2\right) \tag{9}$$

under constraints (2), (3), (4),(5), (6) and (8).

3 Neural network as a fast solver for linear quadratic programs

The basic idea for solving an optimization problem using a tailored neural network is to make sure that the neural network will converge asymptotically at a fast rate and that the equilibrium point of the neural network will correspond effectively to the solution of the original optimization problem. In 1986, Tank and Hopfield introduced a linear programming neural network solver realized with an analogic circuit which appeared to be well suited for applications requiring on-line solutions [5]. After this first successful attempt, many neural network models for solving linear and quadratic programming problems have been proposed in the literature. For a review see [6,7] and an application see [10].

According to the relationship between the states of the neural network and the values of primal and dual decision variables, it is possible to divide the existing recurrent neural network for solving linear and quadratic programming problems into three classes: primal neural network, primal-dual neural network, and dual neural network.

In the present case, the mathematical programming problem presents inequality constraints as well as bounding limits. The adoption of a primal-dual neural network

leads to add various slack variables, turning the size of network larger. This primal-dual neural network is built such as global convergence is guaranteed while the convergence speed can be adjusted by choosing an adequate value for its learning parameter [8]. To display the structure of the linear quadratic neural network solver, a a general linear-quadratic programming problem is parametrized as follows:

$$\min \quad f(\underline{\delta}) = \frac{1}{2}\underline{\delta}^T Q \underline{\delta} + \underline{c}^T \underline{\delta} \tag{10}$$

$$\text{s.t.} \quad h(\underline{\delta}) = J\underline{\delta} - \underline{d} = \underline{0} \tag{11.1}$$

$$g(\underline{\delta}) = A\underline{\delta} - \underline{b} \le \underline{0} \tag{11.2}$$

$$\underline{\xi}^- \le \underline{\delta} \le \underline{\xi}^+ \tag{11.3}$$

$\underline{\delta}$ is the decision vector, representing in our case the positions of the swarm entities. Matrix Q is assumed symmetric positive semi-definite which allows to handle in a similar way linear quadratic and linear programming problems.

Once constraints (11.1), (11.2) and (11.3) are feasible, at least one optimal solution $\underline{\delta}^*$ will meet the Karush-Kuhn-Tucker optimality conditions (KKT) [9]. Then (1, 2) can be turned equivalent to the following set of linear variational inequalities:

$$\left(\underline{y} - \underline{y}^*\right)^T \left(H\underline{y}^* + \underline{p}\right) \ge \mathbf{0} \quad \forall \underline{y} \in \Omega \tag{12}$$

with the primal-dual variables $\underline{y} = \left[\underline{\delta}^T \quad \underline{u}^T \quad \underline{v}^T\right]^T$. Then the problem is to find a solution vector $\underline{y}^*$. Its feasible region Ω and its lower/ upper limits are given by:

$$\Omega := \left\{\underline{y} \middle| \underline{\varsigma}^- \le \underline{y} \le \underline{\varsigma}^+\right\} \quad \underline{\varsigma}^- = \left[\underline{\xi}^- \quad -\underline{\omega}^+ \quad \underline{0}\right]^T \text{ and } \underline{\varsigma}^+ = \left[\underline{\xi}^+ \quad \underline{\omega}^+ \quad \underline{\omega}^+\right]^T \tag{14}$$

Here $\underline{\omega}^+$ has an appropriate dimension and each of its entries is chosen to be sufficiently large to replace $+\infty$ numerically. The coefficients are defined as:

$$\underline{\rho} = \left[\underline{c}^T \quad -\underline{d}^T \quad \underline{b}^T\right]^T \text{ and } H = \begin{bmatrix} Q & -J^T & A^T \\ J & \underline{0} & \underline{0} \\ -A & \underline{0} & \underline{0} \end{bmatrix} \tag{15}$$

Then the neural network model which solves (10) with (11) is given by:

$$\frac{d\underline{y}}{dt} = \lambda\left(E + H^T\right)\left\{P_\Omega\left(\underline{y} - \left(H\underline{y} + \underline{\rho}\right)\right) - \underline{y}\right\} \tag{16}$$

where λ is a positive learning parameter which can be used to adjust the convergence speed of the network, E is an identity matrix, $P_\Omega[\cdot]$ is a piecewise-linear function defined as:

$$P_\Omega[y_i] = \begin{cases} \varsigma_i^-, & \text{if } x_i \le \varsigma_i^- \\ \varsigma_i^+, & \text{if } x_i \ge \varsigma_i^+ \\ y_i, & \text{otherwise} \end{cases} \tag{17}$$

4 Numerical example

Here we present a simulation of a small pursuer-relay-target system composed of two pursuers, two targets and two relays. The communication radius has been set to 15m and the agents move over an area of 200m². The respective speeds of pursuers and targets have been set to 10 m/s and 5 m/s. The neural network parameters have been chosen such as 10^{10} replaces $+\infty$ in (3), $\lambda = 10^5$, $d = 20$ m, $\delta = 5$ m, and $r = 10$ m. The targets have been deployed initially at positions (-50m,-50m) and (50m, 50m). The arborescence including the pursuers (P_1 and P_2) and the relays (R_1 and R_2,) is represented as $P_1 - R_1 - R_2 - P_2$.

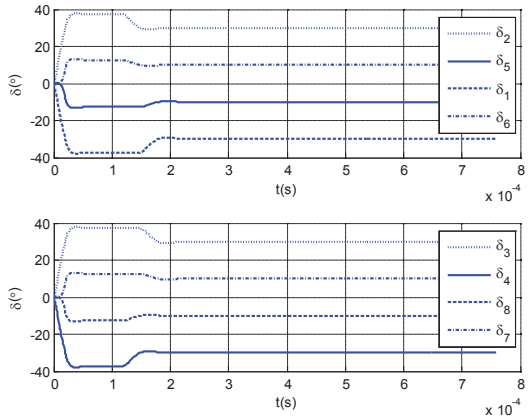

Fig. 2 Convergent behavior of the neural network.

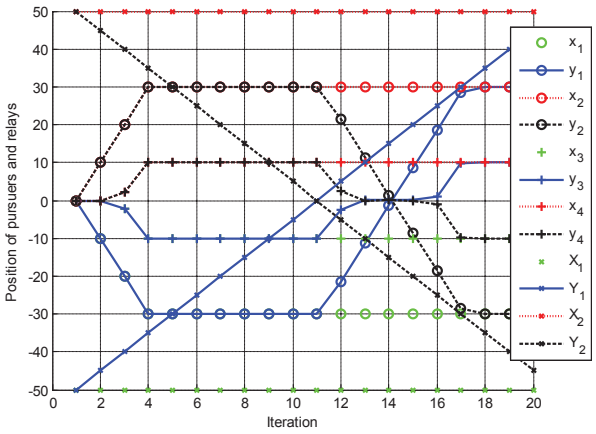

Fig. 3 Positions of entities over 20 periods.

Figure 2 shows that in general the neural network converges after an operation of less than 0. 2s. Figure 3 displays the trajectories of targets and pursuers over time.

The dotted lines indicate the positions of targets while solid and dashed lines represent respectively the positions of pursuers and relays. The trajectories of target 1 and 2 have been chosen as vertical lines with respectively steady increasing and decreasing positions at rate δ m/stage.

5 Final remarks

In this paper, we have presented a solution to the problem of positioning pursuers and relays over a limited free area where targets are moving. This has led to the formulation of a linear quadratic programming model, under connectivity constraints. In order to solve this problem at an efficient rate compatible with the pursuit of the targets, a primal-dual neural network has been built and used as a solver. Moreover, a numerical example has been presented which displays the feasibility of the proposed solution strategy.

Future improvements should consider first the optimization of the arborescence by including new relay nodes as the pursuers are involved in the exploration of larger areas. Then decentralised schemes will be proposed to pursue targets as well as maintaining dynamically connectivity between pursuers. This paper displays the result of a preliminary study towards this objective.

References

[1] M. M. Zavlanos, A. Jadbabaie and G. J. Pappas. Flocking while Preserving Network connectivity, In *Proceedings of the 46th IEEE Conference on Decision and Control*, New Orleans, LA, USA. 2007.

[2] O. Tekdas, P. A. Plonski, N. Karnad and V. Isler. Maintaining Connectivity in Environments with Obstacles. In Robocomm'07. Proceedings of the 1st international conference on Robot Communication and Coordination. 2007.

[3] Stump, E. Jadbabaie and A. Kumar, V. Connectivity management in mobile robot teams, In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on* , pp.1525-1530, 19-23 May 2008.

[4] Tekdas, O.; Plonski, P.A.; Karnad, N.; Isler, V.; Maintaining connectivity in environments with obstacles, *Robotics and Automation (ICRA), 2010 IEEE International Conference on* , pp.1952-1957, 3-7 May 2010.

[5] D. W. Tank and J. J. Hopfield. Simple 'Neural' Optimization Networks: An A/D Converter, Signal Decision Circuit, and a Linear Programming Circuit. *IEEE Transactions on Circuits and Systems*, 5(33): 533–541, 1986.

[6] Y. Xia and J. Wang. Reccurent Neural Networks for Optimiztion: the State of the Art. In *Reurrent neural networks: design and applications*, Chapter 2, CRC Press, 2001.

[7] M. S. Kamel and Y. Xia. Cooperative recurrent modular neural networks for constrained optimization: a survey of models and applications. *Cognitive Neurodynamics*, 3(1):47-81, Mar. 2009.

[8] Y. Zhang. On the LVI-based Primal-Dual Neural Network for Solving Online Linear and Quadratic Programming Problems. In *American Control Conference*, pages 1351-1356, Porland, OR, USA, 2005.

[9] D. G. Luenberger and Y. Ye. *Linear and Nonlinear Programming*. Springer, 2008.

[10] L. Zhong and F. Mora-Camino, Neural Networks based Aircraft Fault Tolerant Control *14th AIAA/ISSMO Conference*, Indiana, USA, 2012.