Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

Cooperative Task Assignment Using Dynamic Ranking

A. Bracci * **M. Innocenti** ** **L. Pollini** ***

* *Department of Electrical Systems and Automation, University of Pisa, 56126 Pisa, Italy; (e-mail: bracci@dsea.unipi.it).*
** *Department of Electrical Systems and Automation, University of Pisa, 56126 Pisa, Italy (e-mail: minnoce@dsea.unipi.it)*
*** *Department of Electrical Systems and Automation, University of Pisa, 56126 Pisa, Italy (e-mail: lpollini@dsea.unipi.it)*

Abstract: The paper presents a novel approach to the decentralized task assignment for multiple cooperative unmanned air systems, in a multiple target-multiple task environment. The vehicles (or agents) may have complete or partial a priori information about the targets that populate the scenario. Each vehicle autonomously computes the cost for servicing each task available at each target using a path planning algorithm, taking into account obstacles, pop-up threats, and weights the total path cost including potential risk areas. Vehicles assign an initial ranking to each task, and then exchange their ranking information with the others. Each agent then updates the ranking of its tasks using a non linear dynamic programming algorithm that is proven to be stable and to converge to an equilibrium point. The ranking dynamics is initially formulated as a continuous time system, and then time-discretized depending on available data, and transmission rate among the network. Stability of the network and independence of steady state values from the data rate are proved analytically. Current studies are directed towards the effect of communication delays. The validity and performance of the proposed method are verified via extensive numerical simulation, and compared with alternate techniques such as an optimal MILP based integrated task assignment and path planning solver.

Keywords: Co-operative control; Decentralized control; Dynamic systems; Discrete systems; Tasks.

NOTATION

n	number of agents/vehicles
m	number of tasks
$w_{i,j}$	generic weight
$c_{i,j}$	simple task cost
$b_{i,j}$	simple task benefit
$C_{i,j}$	task-to-task benefit
$B_{i,j}$	total task benefit
T	sampling time
$\mathbf{1}$	ones matrix

1. INTRODUCTION

Cooperative control among a team of agent is a complex problem of optimization involving the path-planning and the task-assignment issues. While the path-planning can be efficiently solved by many existing procedures (Hershberger and Suri [1999], Huang and Chung [2004], Kallmann et al. [2003]), the task assignment is a much harder problem and finding the optimal solution is known to be *NP-hard*; for this reason many alternative approaches have been proposed in literature. An optimal procedure based on *Mixed Integer Linear Programming* (MILP) have been employed to solve the path planning and the task assignment problems at the same time (How et al. [2001], How et al. [2004]). This method has the advantage of producing the optimal solution w.r.t. a specific cost index that can be modified depending on the objectives of the mission. On the other hand, the MILP approaches to the cooperative control problem reveal too much computationally expensive even in receding-horizon realizations. Moreover the MILP approach is intrisically centralized and not dynamic. These problems prevent the use of MILP for real-time implementations.

Another common approach is to divide the path planning and the task assignment problems. This approach allows to focus on the task assignment assuming that the path planning is solved at an earlier step. The present work follows the latter approach focusing on the cooperative task assignment. Many procedures are proposed in literature, the majority take as an input the cost of each task to be performed and then try to assign the vehicles while obtaining the smallest possible total cost. A simple task-assignment procedure is based on the *Hungarian Algorithm* (Munkres [1957]) that minimizes the total sum of the costs of the assigned tasks. From a control point of view this procedure has the drawback of being centralized and not dynamic and hence it is not applicable in real-time implementations. Another task-assignment procedure found in literature is based on *Auctioning* (Bertsekas and Castanon [1993], Ahmed et al. [2005]). Following the *Auctioning* approach, each agent can act as the *auctioneer*

for a given task while the other agents act as *bidders* following a different dynamics. The *price* of a given task is increases and, after some steps, the *auctioneer* announces the winners of the auction. *Auctioning* has the advantage of being a dynamic procedure but the decentralization is not full because the *auctioneer* act as a central node for the rest of the team. From a computational point of view the agents must perform both the *auctioneer* and the *bidder* policy at the same time, because there can be many simultaneous *auctions*.

In the present work a novel approach to decentralized task-assignment is proposed and it is based on a dynamic task ranking (*DTR*) procedure. The remaining of the paper is organized as follows: in section §2 the generale problem is defined; then in section §3 the *dynamic task ranking* is presented together with many properties. In section §4 the discrete version of the procedure and the implementation issues are presented. Finally in §5 some examples concludes the paper.

2. PROBLEM DEFINITION

The problem addressed is the task assignment among a group of autonomous agents. It is assumed that each agent is capable of calculating the cost c_k it experiences to accomplish any given task k. If an agent is not capable of performing a particular task, then the associated cost is set to infinity. In this work benefits are used instead of costs. Benefits b_k are defined as follows:

$$b_k = \frac{1}{c_k};\ k = 1, 2, \ldots, n \tag{1}$$

Since the costs c_k are non-negative values, then the benefits are non-negative as well. In the remainder it is assumed that $b_{i,j}$ is the benefit that the i^{th} agent has to accomplish the j^{th} task, following (1). Definition (1) takes care of the agent-task costs only, neglecting the task-task costs. This is a crucial point since, as it will be seen later, using a simple control architecture, all the agents can have a more larger view of the entire scenario. In order to consider the task-task costs the total benefit $B_{i,j}$ of a given task j for a given agent i is defined as follows:

$$B_{i,j} = b_{i,j} + \sum_{t=1, t\neq j}^{m} C_{j,t} \tag{2}$$

where $C_{j,t}$ is the benefit that a generic agent has to accomplish the t^{th} task after the completion of the j^{th} task, defined in analogy with (1). The benefit of a task then depends not only on the single task itself, but also on the other tasks configuration, that is the benefit of servicing the other tasks after the completion of the given task. In this view, if many tasks are *near* (that is, the relative task-task costs are low) it means that if an agent accomplished one of those task, then it will be favored to accomplish the others.

3. ASSIGNMENT PROCEDURE VIA DYNAMIC RANKING

In this subsection the decision dynamics of the of the agents are defined. Each dynamic is fully decentralized and it is performed by a single agent. It is assumed that the total benefits $B_{i,j}$ are positive constant values. Let $w_{i,j}$ be the generic weight that task j has on the agent i. The weights of each agent must verify the following relations:

$$0 < w_{i,j} \leq 1 \tag{3}$$

$$\sum_{j=1}^{m} w_{i,j} \leq 1 \tag{4}$$

The entire set of weights W_v of an agent v is defined as follows:

$$W_v = \{w_{i,j} \mid i = v, j = 1 \ldots m\} \tag{5}$$

The decision dynamics of each agent are conceptually divided into two parts: the first can be called *selfish* while the second can be named *cooperative*. The former is a linear system in the weights variables:

$$\dot{w}^S_{v,t} = (1 - \sum_{j=1}^{m} w_{v,j}) B_{v,t} \tag{6}$$

The latter is a nonlinear system depending on the weights and benefits values of the other agents:

$$\dot{w}^C_{v,t} = -\lambda_v \left(\sum_{i=1, i\neq v}^{n} B_{i,t} w_{i,t} \right) w_{v,t} \tag{7}$$

Adding (6) and (7), yields the resulting weight derivative:

$$\dot{w}_{v,t} = \left(1 - \sum_{j=1}^{m} w_{v,j} \right) B_{v,t} - \lambda_v \left(\sum_{i=1, i\neq v}^{n} B_{i,t} w_{i,t} \right) w_{v,t} \tag{8}$$

3.1 Well Posedness

Considering all the weights of all the agents the total dynamics can be written as follows:

$$\dot{W} = f(W) \tag{9}$$

where:

$$W = \{w_{i,j} \mid i = 1 \ldots n; j = 1 \ldots m\} \tag{10}$$

The dynamics expressed in (9) is a nonlinear differential equation and, given some initial values, the standard Cauchy problem is obtained:

$$\begin{cases} \dot{W} = f(W) \\ W(t_0) = V_0 \end{cases} \tag{11}$$

where:

$$\begin{cases} f : \Omega \to \Re^{nm} \\ W, V_0 \in \Omega \\ \Omega = \{x \in \Re^{nm} \mid x_{v,t} > 0, \sum_{j=1}^{m} x_{v,t} \leq 1, \\ \qquad\qquad \forall v = 1 \ldots n, t = 1 \ldots m\} \end{cases} \tag{12}$$

Definition 1. The problem given by (11) and (12) is said to be well posed if the solution exists and it is unique.

In the remaining of this section the well-posedness of the problem is analyzed. To this end we state the following theorem:

Theorem 2. The problem defined in (11) together with (12) is well posed.

Proof Let $f_{v,t} = \dot{w}_{v,t}$ be the generic weight derivative, then the generic entry of the Jacobian matrix is given by the following relations:

$$\frac{\partial f_{v,t}}{\partial w_{i,j}} = \begin{cases} -B_{i,t} & \text{if } i = v, j \neq t \\ -B_{i,t} - \lambda_v s_{v,t} & \text{if } i = v, j = t \\ -\lambda_v w_{v,t} B_{i,t} & \text{if } i \neq v \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

where:

$$s_{v,t} = \sum_{i=1, i\neq v}^{n} B_{i,t} w_{i,t} \tag{14}$$

From (13) and since the $B_{i,j}$, $w_{i,j}$ and λ_v are bounded, then the Jacobian matrix has bounded norm. This is a sufficient condition (together with the continuity of $f(W)$) to the Cauchy-Lipschitz theorem to hold, and then for any given initial point V_0 the solution of (11) exists and it is unique. As a consequence, the stated problem is *well-posed*.

3.2 Stability

In this subsection the stability properties of the dynamic ranking are presented.

Proposition 3. Given an initial condition $V_0 \in \Omega$ and $V_0 \notin \partial\Omega$, then the time evolution of the state variables is fully inside Ω.

Proof The proof of the previous proposition is made for exhaustion. Assume that there exist a time $\bar{t}$ such that $W(\bar{t}) \notin \Omega$. Since the nonlinear system (15) does not present discontinuities, and since the initial values are strictly inside Ω, then there exist a time t_C such that $t_C < \bar{t}$ and $W(t_C) \in \partial\Omega$. In this case, two different situations can be verified:

- Any of the state variables $w_{v,t} = 0$. In this case the nonlinear part of the dynamics is multiplied by zero and hence $\dot{w}_{v,t}$ is non-negative.
- The sum $\sum_{j=1}^{m} w_{v,j}$ is equal to one. In this case, the state variables derivatives of vehicle v are affected by the *cooperative* part only, and hence $\dot{w}_{v,j} \leq 0$ for all j

The previous two conditions mean that if $W(t_C) \in \partial\Omega$ then $\dot{W}$ has a direction which is inside Ω thus demonstrating the proposition.

3.3 Equilibrium Point

Consider an equilibrium point $\bar{W}$ of (11), which is found by solving the following equation:

$$\dot{W} = f(\bar{W}) = 0 \tag{15}$$

denote with $\bar{W}$ the solution of (15) and assume that $\bar{W} \in \Omega$. From the previous subsection it follows that $\bar{W}$ is at least marginally stable. At the present time, asymptotic stability of the resulting dynamics is not proven though simulations have shown this desirable property in all the cases. In addition the equilibrium point of the simulations seems to be the only attractor point in the domain Ω; this nice property has been found by solving nonlinear system (15): many initial guesses of the solving procedure have been chosen randomly in the domain Ω and in all the cases the convergence point has been the same. Moreover this point is the equilibrium point of the dynamic simulation of (9).

3.4 Assignment Properties

Based on the considerations in §3.1 and §3.2, we can state that the weights dynamics evolves into the domain Ω for each agent and then each $w_{v,t}$ is surely into the interval $(0, 1]$. As a consequence of this, it is possible to establish a ranking among the weights of each agent. The largest weight of an agent represents its *best-task*. Once the full dynamic has converged, the resulting assignment is identified by the largest weight of each agent. It is worth noticing that, contrary to other existing task assignment procedures (such as the *Auctioning*, (Ahmed et al. [2005]) for instance), there is not a *master* agent and the team self-organizes in order to achieve a correct, and hopefully optimal, assignment. At present time, simulations have shown that the final weights configuration is such that each agent has only one best target, that is, the maximum weight value is assumed by only one variable.

4. IMPLEMENTATION ISSUES

In this section the physical realization of the dynamic ranking is analyzed. The first step is to obtain an appropriate discrete dynamics deriving from (8) considering the intrinsic system delays due to the sampling time T. Then the discrete dynamics properties are studied in order to obtain relationship between the continuous and the discrete system.

4.1 Dynamics Discretization

The main property of (8) is the separation between the *selfish* and the *cooperative* part that can be viewed respectively as *internal* and *external* dynamics (with respect to a single agent). The *internal* dynamics evolves with the *internal* variables only, while the other variables affects the *cooperative* part only. This is a very useful property because, in the view of a real-time implementation, during the intersampling time the *extern* variables (14) remain constant. The realization of (8) during the intersampling time then becomes:

$$\dot{w}_v(t) = A_v^c w_v(t) + B_v^c \tag{16}$$

where:

$$w_v(t) = [\, w_{v,1} \;\; w_{v,2} \;\; \dots \;\; w_{v,m} \,]^T \tag{17}$$

$$A_v^c = -\text{diag}(\, B_{v,1} \;\; B_{v,2} \;\; \dots \;\; B_{v,m} \,)\mathbf{1} - \lambda_v \text{diag}(\, s_{v,1} \;\; s_{v,2} \;\; \dots s_{v,m} \,) \tag{18}$$

$$B_v^c = [\, B_{v,1} \;\; B_{v,2} \;\; \dots \;\; B_{v,m} \,]^T \tag{19}$$

B_c is a constant vector and the resulting system (16) is then linear and autonomous. Therefore, in order to obtain the correct weights values after a sampling time, *exact discretization* can be employed obtaining:

$$w_v(k+1) = A_v^d w_v(k) + B_v^d \tag{20}$$

with:

$$A_v^d = e^{A_v^c T} \tag{21}$$

$$B_v^d = \int_0^T \left(e^{A_v^c T} B_v^c \right) dt = (A_v^c)^{-1} \left(e^{A_v^c T} - I \right) B_v^c \qquad (22)$$

The closed form of $e^{A_v^c T}$ is not known, it can however be evaluated with very high precision using a robust numerical procedure (see (Moler and Loan [2003]) for a detailed treatment). Since the discrete dynamics is obtained by the exact discretization of the continuous time dynamics then the stability properties presented in §3.2 hold for the discrete system as well.

4.2 Real-Time Issues

In real-time applications each agent may have a different sample time and, in the general case, there is not synchronization between the agents. In this work it is assumed for simplicity that each agent has a fixed sample time T and that all the agents are synchronized. This last hypothesis is reasonable since, in real applications, the vehicles can be synchronized by using the GPS clock. The previous assumptions allow to build the complete weights dynamics by concatenating (21) and (22) for all the agents. The complete discrete system then becomes:

$$w = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} \qquad (23)$$

$$A^d = \text{diag}\left(A_1^d,\ A_2^d,\ \dots\ A_n^d \right) \qquad (24)$$

$$B^d = \begin{bmatrix} B_1^d \\ B_2^d \\ \vdots \\ B_n^d \end{bmatrix} \qquad (25)$$

$$w(k+1) = A^d(w(k))w(k) + B^d \qquad (26)$$

The system (26) is *pseudo-linear* since the matrix A^d depends on the values of $w(k)$. The main property of (26) is that it directly deals with the common real-time problems such as *computation time, data send and receive*, since the selected sampling time T can be chosen such that, during the intersampling period each vehicle is able to perform the requested calculus following (20) then send the resulting values to the other agents, and finally receive the data from the other agents. From §4.1 it is clear that the sample time has no effect on the stability properties thus a great advantage of the dynamic ranking is its direct applicability in real-time situations even using large sampling times. Moreover, though not proven yet, simulations have shown that using the discrete dynamics (26) the steady values of the weights are the same of those obtained from (16). Numerical simulations have also shown that the steady values of the weights converge to a locally stable equilibrium point. In fact, let $\bar{w}$ be the solution of the following nonlinear system:

$$\bar{w} = A^d(\bar{w})\bar{w} + B^d \qquad (27)$$

then the resulting matrix $A^d(\bar{w})$ is such that the greatest eigenvalue is completely inside the unit circle, thus yielding the local asymptotic stability. In the remaining of this section, many consequences of the proposed approach are presented, focusing on the scenario dynamism and the hardware limitations.

4.3 Scenario Dynamism

In section §3 the values of the benefits $B_{i,j}$ are assumed to be constant. However, since the benefits depend on the scenario configuration, the values of $B_{i,j}$ may change during the mission. In order to correctly deal with the benefits variation rate it is necessary that the weights evolution reach the steady values before the benefits can change. In this section an estimate of the maximum admissible benefits variation rate is presented.

From section §4.2 the steady state of the discrete system $\bar{w}$ equals the steady state of the continuous system, then a linearization of the complete system can be employed in the neighborhood of $\bar{w}$. The resulting linear system has a time constant τ which is identified by the slowest eigenvalue δ_{SL} as follows:

$$\tau = \frac{1}{\delta_{SL}} \qquad (28)$$

Then, in the neighborhood of $\bar{w}$ an estimate of the settling time T_S of the entire system is given by the following relation:

$$T_S = 3\tau \qquad (29)$$

In order to let the weights reach their steady state the benefits must remain unchanged for a time greater than T_S. Since the weights evolve in a discrete-time system, then the benefits update can be made after the lowest integer multiple of T that exceeds T_S. If the updated values of the benefits are *near* to the previous ones then the linearization is in its range of validity and hence a quantitative maximum benefits rate can be estimated from (29).

4.4 Maximum Number of Vehicles and Tasks

In section §4 the main properties of the discrete version of the dynamic ranking are presented neglecting the real vehicles capabilities. In this subsection the hardware limitations due to the intra-team communications are considered. From 4.2 it follows that the sample time T has no effects on the stability properties of the global procedure. However the value of T affects the transient of the dynamics before steady state is reached. Simulations have shown that using very large sampling times ($T \geq 50s$) the resulting behaviour, though still simply stable, presents steady oscillations about the equilibrium point. However, using more realistic sampling times ($T \leq 10s$) the dynamic becomes asymptotically stable to the equilibrium point.

Another interesting effect of the sampling time is its influence on the maximum number allowed of agents and tasks in the scenario. Let us consider a standard communication rate of 9600 bit/s (baudrate, $BR = 9600$), and suppose that the data to be transmitted among the vehicles are composed by *doubles* (7). Each *double* is composed by 8 *bytes*, moreover the transmission of each double must be preceded and followed by a bit. For this

reason, the maximum number of doubles that can be transmitted is given by the following relation:

$$D_R = \frac{BR}{10 \times 8} = \frac{9600}{10 \times 8} = 120 \frac{doubles}{s} \tag{30}$$

Each vehicle must transmit m *doubles* and an identification code which can be identified with a double. Then the total data to be transmitted by a single vehicle are $m + 1$. The resulting total time T_T for the transmission of each vehicle must respect the following relationship

$$T_T \geq \frac{m+1}{D_R} \tag{31}$$

Finally, since all the vehicles must transmit and receive into each intersampling time, then the following must hold:

$$\left\lfloor \frac{T}{T_T} \right\rfloor \geq n \tag{32}$$

From (31) and (32) a relationship between n, m, and T is found. Some numerical examples are presented in Table 1.

Table 1. Relation between T, n and m

T	T_T	n	m
1	.1	10	10
1	0.05	20	5
1	0.025	40	2
1	0.2	5	20

5. EXAMPLES

In this section some examples are presented, to show the capabilities of the dynamic task ranking (*DTR* in the following).

Example 1 The first example is a simple obstacle-free scenario, which is represented in figure 1.

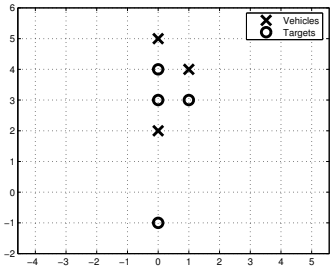

Fig. 1. Scenario Geometry

The three vehicles are near to the three upper targets while the remaining target is far from the others. Assignment procedures such as the *Auctioning* and the *Hungarian Algorithm* would produce the assignment shown in figure 2.

Using the *DTR*, instead, the assignment is shown in figure 3, while the weights time response of vehicle V_1 is represented in figure 4.

It is clear from figure 3 and from Table 2 that the total mission completion time is sensibly reduced using the *dynamic task ranking* (in Table 2 the results are compared also with the optimal assignment given by the solution of

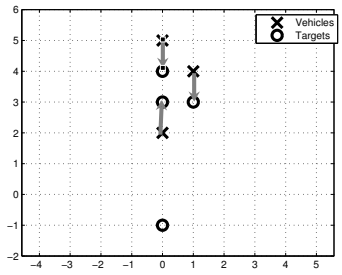

Fig. 2. Hungarian Algorithm/Auctioning Results

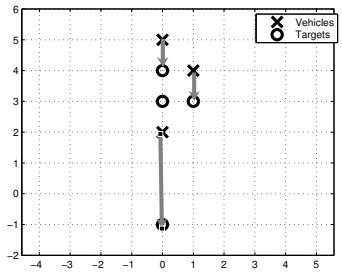

Fig. 3. Dynamic Task Ranking Assignment

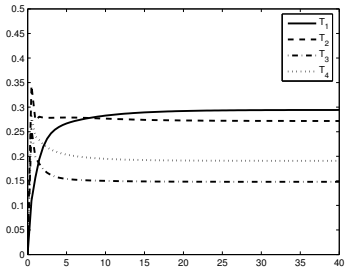

Fig. 4. Weights time response with static benefits of vehicle V_1

the associated *Integer Programming* problem). Moreover, vehicles movement is allowed in order to show the different response whenever the benefits change. It is assumed that each vehicle moves toward its best target, which is identified by its largest weight. The results of the weights dynamics of the first vehicle are represented in figure 5.

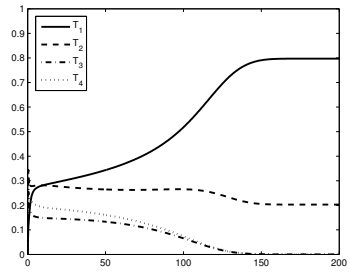

Fig. 5. Weights time response allowing varying benefits of vehicle V_1

From figure 5 it is clear that as the mission evolves, the largest weight tends to increase while the others tends to a lower value.

	J_{time}	J_{fuel}
Int. Prog.	3	6
Hung.	5	7
Auct.	5	7
DTR	3	6

Table 2. Numerical results of Example 1. J_{time} is the mission completion time; J_{fuel} is the team fuel consumption.

Example 2 This example is more complex and it represents a simplification of the Gulf of La Spezia (Italy) area, presented in figure 6.

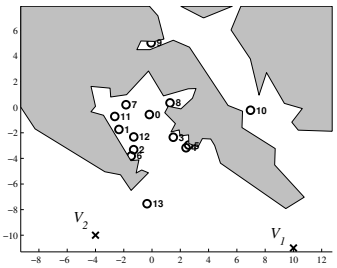

Fig. 6. Gulf of La Spezia (Italy) scenario. X = vehicles; O = Targets; Solid areas = obstacles. Units in km.

There are many concentrated targets, while two targets are far away from the others. Existing assignment procedure (such as *Hungarian Algorithm* and *Auctioning*) would produce the assignment shown in figure 7. While the *DTR* produces a better assignment as presented in figure 8.

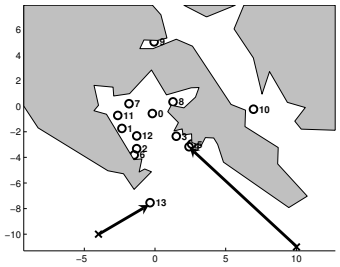

Fig. 7. Hungarian/Auctioning assignment of the *La Spezia* scenario.

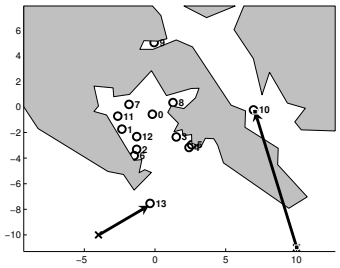

Fig. 8. DTR assignment of the *La Spezia* scenario.

The weights dynamics of a vehicle is represented in figure 9. In this scenario the benefits are varying, this is why the best weight has an increasing behavior until the task is completed. Note that the not-selected tasks decrease their weights as the best task increases.

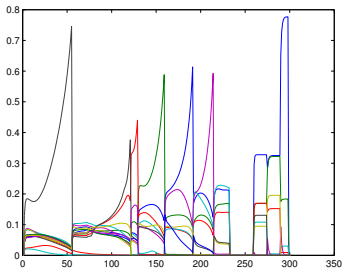

Fig. 9. Weights time response of vehicle V_1

6. CONCLUSIONS AND FUTURE WORKS

Dynamic task ranking is a novel approach to address the decentralized cooperative task assignment problem. The procedure requires a low computational cost and is fully decentralized. Stability properties have been proven for both the continuous and the discrete time version of the control law. Simulations have shown the asymptotic stability to an equilibrium point in all cases. This equilibrium point can be found also by solving the nonlinear system associated to the weights dynamics. Though not proven yet, it appears that the equilibrium point is the only attractor of the dynamics, starting from any admissible starting point.

REFERENCES

A. Ahmed, A. Patel, T. Brown, M. Ham, M. Jang, and G. Agha. Task assignment for a physical agent team via a dynamic forward/reverse auction mechanism. *International Conference of Integration of Knowledge Intensive Multi-Agent Systems KIMAS '05: Modeling, Evolutions and Engineering, 18 - 21 April*, 2005.

D. P. Bertsekas and D. A. Castanon. A forward/reverse auction algorithm for asymmetric assignment problems. *Technical Report Lids-P-2159, MIT*, 1993.

J. Hershberger and S. Suri. An optimal algorithm for euclidean shortest paths in the plane. *SIAM Journal of Computation*, 28(6):2215–2256, 1999.

J. How, T. Shouwenaars, B. De Moor, and E. Feron. Mixed integer linear programming for multi-vehicle path planning. *European Control Conference 2001, Porto, Portugal*, pages 2603–2608, 2001.

J. How, E. King, and Y. Kuwata. Flight demonstrations of cooperative control for uav teams. *AIAA 3rd Unmanned Unlimited Technical Conference, Workshop and Exhibit. 20-23 September 2004, Chicago, Illinois*, 2004.

H-P. Huang and S-Y. Chung. Dynamic visibility graph for path planning. *proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, September 28 - October 2, 2004, Sendai, Japan*, 2004.

M. Kallmann, H. Bieri, and D. Thalmann. Fully dynamic constrained delaunay triangulations. *Geometric Modelling for Scientific Visualization, Heidelberg, Germany*, 2003.

C. Moler and C. Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Review*, 45(1):3–49, 2003.

J. Munkres. Algorithms for the assignment and transportation problems. *Journal of SIAM*, 5(1):32–38, 1957.