Distributed networked control systems (D-NCS) are vulnerable to various network attacks when the network is not secured; thus, D-NCS must be well protected with security mechanisms (e.g., cryptography), which may adversely affect the dynamic performance of the D-NCS because of limited system resources. This paper addresses the tradeoff between D-NCS security and its real-time performance and uses the Intelligent Space (iSpace) for illustration. A tradeoff model for a system’s dynamic performance and its security is presented. This model can be used to allocate system resources to provide sufficient protection and to satisfy the D-NCS’s real-time dynamic performance requirements simultaneously. Then, the paper proposes a paradigm of the performance-security tradeoff optimization based on the coevolutionary genetic algorithm (CGA) for D-NCS. A Simulink-based test-bed is implemented to illustrate the effectiveness of this paradigm. The results of the simulation show that the CGA can efficiently find the optimal values in a performance-security tradeoff model for D-NCS.

Index Terms—Coevolutionary genetic algorithm (CGA), distributed networked control systems (D-NCS), iSpace.

I. INTRODUCTION

Distributed networked control systems (D-NCS) are spatially distributed systems in which the control loops are closed through a real-time network. This approach integrates the computing and communications capabilities with the monitoring and control of entities in the physical world. These systems are usually comprised of a set of networked agents that include distributed sensors, actuators, controllers, and a communications network.

With the rapid advancements in the Internet, embedded systems, and wireless communications technologies in recent years, research on D-NCS has been gaining popularity because of their high potential for widespread applications [1], [2], such as monitoring and operations for manufacturing plants, space projects, defense systems, robot navigations, nursing homes, traffic management, and many more.

Many of these applications are time-sensitive, data-sensitive, and safety-critical. The potential consequences of compromising the D-NCS can be devastating to public health and safety, national security, and the economy. Compromised D-NCS can, and have, led to extensive cascading power outages, dangerous toxic chemical releases, and explosions [3]. It is, therefore, important to implement D-NCS with security controls that make reliable, safe, and flexible performance possible. In addition, a wireless medium is easily susceptible to interception, which may pose increasing concerns about the security of communications on the D-NCS. To reduce operational costs and improve performance, D-NCS were transitioned to less expensive standardized technologies, operating systems, and protocols that are currently prevalent (e.g., on the Internet). As a result, real-time monitoring and control information is readily and easily accessible to a large number of people on the Internet. This also increases the vulnerability of the D-NCS to malicious network attacks. Thus, data sharing and communications security are among the main concerns in D-NCS, considering their time-sensitive and data-sensitive applications. It is critical to protect transmitted data from unauthorized access and modification in the D-NCS’s communications channels.

Although D-NCS with security, from a control system’s perspective, are still in their infancy, many D-NCS have been well protected by different levels of security mechanisms [4], [5]. However, adding more security may adversely affect system performance because of limited system resources. The impact of the security mechanisms on system performance has not been addressed thoroughly. Security requirements are often in competition with performance requirements, such as real-time dynamic performance that is limited by system resources [6] and the extra time delay imposed by additional security. Thus, there is a tradeoff between the D-NCS’s performance and its security measures.

Determining how to achieve the optimized balance between performance and security on D-NCS is an open question. The coevolutionary paradigm—inspired by the reciprocal evolutionary change driven by cooperative and competitive interaction among different species—has recently been extended successfully to multiobjective (MO) optimization [7]. As a fast-developing optimization algorithm, the coevolutionary genetic algorithm (CGA) is an extension of conventional evolutionary algorithms. It models an ecosystem consisting of two or more species. Multiple species in the ecosystem coevolve and interact with each other, resulting in continuous evolution of the ecosystem [8]. The CGA is noted by its fast convergence while maintaining a good diversity of solutions in EAs [9]. It is
fairly immune to the local minima and nonlinear nature of the optimization problems being considered in this paper. Thus, the CGA is a prime candidate to solve the performance and security tradeoff for D-NCS.

In this paper, we address the issue of the tradeoff between D-NCS security and the system’s real-time performance. Since most of the current effort for protecting D-NCS has been accomplished with prevention mechanisms (e.g., cryptography), this paper focuses on the confidentiality aspect of security service using secret key cryptography. Our goal is to develop an effective approach to model and optimize this trade-off. We, first, identify and define the performance and security tradeoff problem of D-NCS; second, review related work that has been done in this field; and third, propose a tradeoff model for performance and security in D-NCS. We also suggest an optimization approach for a multiagent performance and security tradeoff based on the CGA. The proposed tradeoff model can be used to simultaneously adjust system resources to provide sufficient protection and satisfy real-time performance requirements for D-NCS, while the CGA is an effective approach for this tradeoff analysis and optimization.

II. RELATED WORK

Since the traditional D-NCS without security protection is vulnerable to various security attacks [10], developing security mechanisms for D-NCS has turned out to be a research hotspot, giving rise to many topics, such as performance assessments for D-NCS with security, novel network architectures to support security for D-NCS, and novel intrusion detection schemes for D-NCS [11].

Dzung et al. [12] gave an overview of Information Technology (IT) security issues in industrial automation based on open communication systems and explained various countermeasures. Cardenas et al. [13] identified and defined the problem of secure control in cyber-physical systems (CPS) and proposed a set of challenges that need to be addressed to improve the CPS’s ability to survive. Kim et al. [14] also gave an overview on the challenges and ongoing efforts in the field of cyber-physical security with specific emphasis on a smart grid infrastructure.

Mukherjee and Gupta [15] established a criticality response modeling (CRM) framework to ensure that the networked control system has criticality-awareness—the ability of the system to respond to unusual situations. Xu et al. [16] developed a core architecture to address the collaborative control issues of distributed device networks under open and dynamic environments by adopting policy-based network security technologies and extensible markup language (XML) processing technologies. Creery and Byres [17] presented methods to determine and reduce the vulnerability of D-NCS to unintended and malicious intrusions for an industrial plant.

Lately, there also has been an increasing concern about protecting the distributed control algorithms (e.g., consensus algorithms) from malicious cyber attacks on the D-NCS. Pasqualetti et al. [18] first introduced the problem of detecting and identifying misbehaving agents in a linear consensus network with a solution for the case of a single faulty-agent. Sundaram and Hadjicostis [19] extended and improved the results along these routes by providing one policy that k malicious agents can follow to prevent some of the nodes of a 2k-connected network from computing the desired function of the initial state or from reaching an agreement. Teixeira et al. [20] proposed a distributed scheme to detect and isolate the cyber attacks in the communication network of the D-NCS using observers and discussed how to reduce the number of observer nodes while maintaining the coverage of the entire network.

Most of the works referenced above deal with the security issues of the D-NCS from the prevention and detection perspectives. However, there is also a growing demand for studying the impact of the security additions on the systems’ performance and the tradeoff between them. Gupta et al. [21] characterized the D-NCS application on the basis of the security’s effect on its performance and mapped the added security features to an increased time delay in the system to show this tradeoff for a path-tracking application. A tradeoff model for conventional networked control systems has been described by Zeng and Chow in [22]. A set of quantitative performance and security metrics have also been developed and combined in a tradeoff objective function. Zeng [23] has also shown a successful two-agent case of optimizing the tradeoff between the system dynamic performance and on-demand security on a networked DC Motor system using CGA. However, D-NCS often have more than two agents. Most of these systems are multiagent systems composed of several distributed sensors, controllers, and actuators, and all of these components are connected over a network. Thus, in this paper, we use one kind of D-NCS—the Intelligent Space (iSpace)—as an example of how to analyze and optimize the tradeoff between system dynamic performance and on-demand security. A quantitative performance-security tradeoff model for D-NCS using iSpace as an illustration is presented and a paradigm of performance and security tradeoff optimization based on CGA is proposed as well.

The remaining sections are organized as follows: Section III provides the description of the iSpace system. Section IV presents the performance-security tradeoff model for the iSpace system. Section V describes the paradigm of the CGA optimization process for the D-NCS tradeoff model. The implementation of a test-bed and the simulation result analysis are presented in Section VI. Section VII concludes the paper and discusses the future work.

III. iSPACE SYSTEM DESCRIPTION

iSpace at the Advanced Diagnosis, Automation, and Control (ADAC) Lab at North Carolina State University (NCSU) has been implemented to perform basic research and education on time-sensitive and secure D-NCS with hardware-in-the-loop fast-prototyping capabilities. iSpace is a network-based integrated navigation system with different modules combined together to guide several unmanned ground vehicles (UGVs) from one point to another where the navigational intelligence comes from remote controllers. It includes a network of distributed sensors (cameras and encoders) and multiple actuators (UGVs). iSpace aggregates the status of the entire space from each agent’s sensory information and responds intelligently to the system’s goals. iSpace has several major components: distributed sensors
and actuators, hybrid hierarchical and distributed controllers, the system’s security manager, and the communication network. The system structure is shown in Fig. 1.

The D-NCS in this paper is composed of multiple UGVs controlled by iSpace that move between locations tracking a path that avoids obstacles in a secured network environment. Thus, UGVs need to track the path and reach their destination efficiently while the security manager has to protect the whole system from malicious network attacks. However, the system resources (network bandwidth) are limited, so the UGVs and system security manager are actually competing for the same resources to achieve their goals.

A. UGV Dynamics

Each UGV has a differential drive with two driving wheels and one caster wheel, as in Fig. 2. The dynamics of the UGV can be described using the kinematic model

\[
\begin{bmatrix}
\dot{v}_{\text{ref}} \\
\dot{\omega}_{\text{ref}}
\end{bmatrix} = \begin{bmatrix}
\frac{\rho}{W} & \frac{\rho}{W} \\
\frac{\rho}{W} & -\frac{\rho}{W}
\end{bmatrix} \begin{bmatrix}
\omega_r \\
\omega_l
\end{bmatrix}
\]  

(1)

where \(v_{\text{ref}}\) is the UGV’s linear velocity, \(\omega_{\text{ref}}\) is its angular velocity, \(\rho\) is the radius of the drive wheels, and \(W\) is the distance between the drive wheels. \(\omega_r\) and \(\omega_l\) represent the angular velocities of the right and left drive wheels, respectively.

The UGV steers by driving the wheels at different speeds determined by solving (1) where \(v_{\text{ref}}\) and \(\omega_{\text{ref}}\) are given in the input reference command

\[
\begin{bmatrix}
\omega_{R,\text{ref}} \\
\omega_{L,\text{ref}}
\end{bmatrix} = \begin{bmatrix}
\frac{1}{\rho} & \frac{W}{\rho} \\
\frac{1}{\rho} & -\frac{W}{\rho}
\end{bmatrix} \begin{bmatrix}
v_{\text{ref}}
\end{bmatrix}.
\]  

(2)

Using (2), the input reference command can be utilized to set reference speeds for the individual wheels. A proportional-integral (PI) controller, defined by (3), is used to achieve the reference speed on each wheel

\[
u(t) = K_P e(t) + K_I \int_0^t e(\xi) d\xi
\]  

(3)

where \(K_P\) is the proportional gain, \(K_I\) is the integral gain, and \(e(t)\) is the difference between the actual wheel speeds and the reference wheel speeds as shown in (4)

\[
e(t) = \begin{bmatrix}
\omega_{R,\text{ref}} - \omega_R \\
\omega_{L,\text{ref}} - \omega_L
\end{bmatrix}.
\]  

(4)

The two control parameters in (3) are used to control the three UGV states defined by (5)

\[
a(t) = \begin{bmatrix}
x_w(t) \\
y_w(t) \\
\varphi(t)
\end{bmatrix}
\]  

(5)

where \(x_w(t)\) and \(y_w(t)\) are the UGV coordinates in the world frame and \(\varphi(t)\) is the UGV heading expressed as the angle between the UGV and the positive x axis of the world frame at time \(t\).

B. Path-Tracking Controller

The quadratic curve (QC) path-tracking controller defined in [24], the feedback preprocessor (FP), and the predictive control gain scheduling middleware (PCGSM) are used to control the UGVs. The path-tracking controller calculates a path, \(\mathbf{p}\), defined by a set of consecutive waypoints on the ground for each UGV to follow as \(\mathbf{p} = [p_x \ p_y]\), where \(p_x\) and \(p_y\) are sets of the world frame \(x\) and \(y\) coordinates, respectively. The QC path-tracking algorithm will determine a look-ahead distance based on previous UGV control. A reference point ahead of the UGV is constrained to move along \(\mathbf{p}\). The location of the reference point is determined by finding the nearest point on the path to the UGV and looking ahead on the path a variable distance defined by using the methods in [24].

To track the reference point, the path-tracking controller finds a QC that passes through the reference point and has its vertex located at the UGV. Since the sampling period is sufficiently small, reference velocity \(v_{\text{ref}}\) and turn rate \(\omega_{\text{ref}}\) can be calculated by fitting a circle in the QC [24] as follows:

\[
A = \frac{e_y}{e_x^2}
\]  

(6)

\[
v_{\text{ref}} = \text{sgn}(e_x) \frac{\alpha}{1 + |A|}
\]  

(7)

\[
\omega_{\text{ref}} = 2A v_{\text{ref}}
\]  

(8)

where \(A\) is the QC coefficient and \(\alpha\) is the maximum velocity. Using \(A\) to determine the velocity causes the UGV to slow down...
while turning to prevent error. Since the reference command is a circular trajectory defined by (7) and (8), the path-tracking controller needs to frequently generate new QCs and subsequent circular arcs for the UGV. The UGV will follow the reference’s circular trajectory for a short time so that when all of the circular arcs are aggregated together, they will approximate the QC, which in turn approximates the desired path.

C. Security Manager

To protect iSpace from malicious network attack, this paper focuses on the confidentiality aspect of the security service using secret key cryptography. The secret key algorithm of the Advanced Encryption Standard (AES) is a symmetric cipher considered secure for wireless systems. It works with 128 bits of block size with key sizes of 128 or more. As Electronic Code Book (ECB) is considered to be the fastest mode of operations, it is used frequently in real-time systems. Thus, in this paper, the AES with an ECB mode is integrated with the application to encrypt and decrypt the information flow along the communication network [25]. Thus, the system must provide the necessary bandwidth for the security mechanisms in the communication channels.

D. Bandwidth Allocation

For simplicity, the bandwidth allocation algorithm used in this paper allocates the bandwidth according to the request of each UGV proportionally.

Each UGV should have a feasible safety region along the path. Outside of this region, the UGV may oscillate, out of control and collide with other UGVs. Therefore, UGVs must remain within the safety region to maintain stability. The stability criterion is then defined as follows:

$$\max (e_i(t)) \leq \delta_i(t)$$  \hspace{1cm} (9)

where $\delta_i(t)$ is the safety region around the path and $e_i(t)$ is the distance from the UGV coordinates to the nearest path point for the $i$th UGV. By putting the system dynamics (1) and the error formulation in the optimization problem, the maximum sampling time $\Delta T_{i,max}$ that guarantees the stability of the $i$th UGV is then calculated by solving the following algebraic equation:

$$e_i(t + \Delta T_{i,max}) - \delta_i(t).$$  \hspace{1cm} (10)

Thus, the minimum bandwidth required in bits per second (bps) that guarantees stability (9) is

$$BW_{i,min} = \frac{L \times 8}{\Delta T_{i,max}}$$  \hspace{1cm} (11)

where $L$ is the size of the original data packet in bytes. The bandwidth needed by the necessary security mechanism is

$$BW_{sec} = \sum_{i=1}^{n} \frac{L_{sec} \times 8}{\Delta T_i}$$  \hspace{1cm} (12)

where $L_{sec}$ is the size of the security addition to the encrypted packet in bytes, which depends on the encryption algorithms. $\Delta T_i$ is the actual sampling time of the $i$th UGV.

So, the stability of every loop in the system is guaranteed as long as (13) is satisfied

$$\sum_{i=1}^{n} BW_{i,min} + BW_{sec} \leq BW_{total}$$  \hspace{1cm} (13)

where $n$ is the number of control loops (UGVs). The remaining bandwidth is shared among the control loops (UGVs) by proportionally allocating it according to the request of each UGV, $BW_{i,req}$. The actual bandwidth allocated to each UGV is calculated as follows:

$$BW_i = BW_{i,req} + \left( \frac{BW_{i,req}}{\sum_{k=1}^{n} BW_{k,req}} \left( BW_{total} - \sum_{k=1}^{n} BW_{k,min} - BW_{sec} \right) \right).$$  \hspace{1cm} (14)

IV. TRADEOFF MODEL FORMULATION

This section describes the performance-security tradeoff model of the iSpace system, including the quantitative performance and security metrics and the tradeoff objective function.

A. Performance Metric

To measure the performance of the UGVs in iSpace, the performance metric, $P_i$, has been implemented. $P_i$ is the accumulated error calculated as the area between the desired path and the actual path of the UGVs

$$P_i = \int_{t_{i,1}}^{t_{i,6}} \frac{1}{2} (a_i(t) - r_i(t))^2 \, dt, \quad i = 1, 2, \ldots, n$$  \hspace{1cm} (15)

where $a_i(t)$ is the actual path that the $i$th UGV travels, and $r_i(t)$ is the desired path that the $i$th UGV should track.

B. Security Metric

Existing qualitative metrics classify various security mechanisms to several discrete levels, such as low, medium, and high. Security mechanisms in the higher level can provide better protection than those in the lower levels. But it’s impossible to compare security mechanisms within the same security level. Furthermore, qualitative metrics are too coarse for fine control of the tradeoff between dynamic performance and security in D-NCS. Thus, a quantitative metric that generates a security strength value for each security mechanism is used in this paper and, hence, is more suitable for a quantitative comparison of the security strength of any two security mechanisms [26].

Without considering any shortcut attacks, brute force attack is the only way used to crack the encryption key in this paper. For example, an AES cipher with a key length of 128 bits has $2^{128}$ possible key combinations. Assuming unit complexity for testing one key, the worst-case complexity involved in cracking this 128-bit AES cipher is $2^{128}$. With this assumption, the security level of an encrypted message frame is decided by its encryption key length. Thus, a security measure with respect to brute force attacks is described as $S_{BF}(N) = \log_2(N)$, where
$N$ is the encryption key length [27]. The security metric of the system is then measured by the vulnerability of its encryption algorithms to brute force attackers, as

$$S = \frac{nS_{\text{max}}}{\sum_{i=1}^{n} \log_2 N_i}, \quad i = 1, 2, \ldots, n$$

(16)

where $S_{\text{max}} = \max(\log_2 N_i)$.

C. Tradeoff Objective Function

With the defined performance metric, $P_i$, and the security metric, $S$, the operational requirements can be formulated quantitatively. Thus, we need to add the following constraints:

$$P_{i, l} < P_i < P_{i, u}, \quad S_l < S < S_u$$

(17)

where $P_{i, l}$ and $P_{i, u}$ are the operational lower and upper bounds, respectively, of dynamic performance for the $i$th UGV, $S_l$ and $S_u$ are the operational lower and upper bounds, respectively, of the system security level.

The performance and security metrics allow us to quantitatively calculate how much protection a security mechanism can provide and how much performance will be degraded if we use it. Therefore, we can make the tradeoff decision between performance and security by adjusting the system resource—allocated bandwidth—within the boundaries of the system operational requirements.

When all the performance and security requirements are satisfied, the system can use the remaining available resources to improve performance, security, or both. A tradeoff objective function among the performance metrics and the security metric is formulated as the utility function below:

$$\text{Min } U = w_1 P_1 + w_2 P_2 + \ldots + w_n P_n + w_{n+1} S$$

subject to: $P_{i, l} < P_i < P_{i, u}$, $S_l < S < S_u$,

$$i = 1, 2, \ldots, n.$$

(18)

where $w_j$ ($j = 1, 2, \ldots, n + 1$) are the weighting factors representing the preferences on different metrics. With different applications, the weighting factors can be adjusted accordingly. In this paper, all the factors are equally weighted.

With this tradeoff objective function, the system can compute and choose the best allocated bandwidth for the security manager together with the optimal allocated bandwidth for each UGV in order to optimize the overall system performance and security.

V. TRADEOFF OPTIMIZATION USING THE CGA

This section presents the paradigm of the performance and security tradeoff optimization for iSpace based on the CGA.

A. Coevolutionary Genetic Algorithm

Similar to genetic algorithms inspired by nature, the concept of coevolution—used as the foundation for the CGA—comes from biological observations. Nature is composed of several species that coevolve. Instead of considering a population of similar individuals that represent a global solution as conventional genetic algorithms (GAs) do, the CGA ponders the co-evolution of subpopulations of individuals representing specific parts of the global solution [28].

The pseudo-code of the CGA is shown in Table I, in which the evolution of each species is handled by a standard GA, while the evaluation of an individual from each species is handled through collaboration with representatives from other species.

\begin{table}[h]
\centering
\caption{The Pseudo-Code of the CGA}
\begin{tabular}{|l|}
\hline
\textbf{Algorithm: CGA} \\
\hline
\begin{algorithmic}
\State \textbf{Generation }$k = 0$
\For {for each species $i$ do}
\State \textbf{initialize the species population }$\text{Pop}[i][0]$
\State \text{evaluate fitness of each individual }$\text{Ind}[i][0][n]$ \text{in }$\text{Pop}[i][0]$
\State \text{choose a representative }$\text{Rep}[i][0]$ \text{from }$\text{Pop}[i][0]$
\EndFor
\While {termination condition = false do}
\For {for each species $i$ do}
\State \textbf{reproduction from }$\text{Pop}[i][k]$ \text{to get }$\text{Mate}[i][k]$
\State \text{crossover and mutation from }$\text{Mate}[i][k]$ \text{to get }$\text{Pop}[i][k+1]$
\State \text{evaluate fitness of each individual }$\text{Ind}[i][k+1][n]$ \text{in }$\text{Pop}[i][k+1]$
\State \text{choose a representative }$\text{Rep}[i][k+1]$ \text{from }$\text{Pop}[i][k+1]$
\EndFor
\State \text{end}
\State \text{end}
\end{algorithmic}
\hline
\end{tabular}
\end{table}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig3}
\caption{Performance-security tradeoff model of D-NCS based on the CGA.}
\end{figure}
in Fig. 3, all the agents interact with each other through the environment and form a noncooperative game [29]. Each agent submits its decision variables (BW_{sec} or BW_{req}) to the system environment model and takes its following actions based on the knowledge of itself and the system response BW_i from the environment. Here, the system environment consists of the total bandwidth BW_{Total} the system has. All the agents submit their decision variables to the system environment model at the same time. The system environment model calculates the response BW_i according to the environment dynamics using (14). Then, each agent can calculate its fitness function (P_i or S) with the system response BW_i by using (15) and (16). In this tradeoff model, the performance agents and the security agent compete against each other for bandwidth to achieve their goals—minimize their own fitness functions, where a noncooperative game is formulated. Together, these goals comprise the final tradeoff objective of the whole system.

The schematics of the CGA optimization process of D-NCS is shown in Fig. 4. Each agent is represented by a species (a species means a population of GA in this algorithm) in the ecosystem shown in Fig. 4. Each species evolves a bundle of individuals that represent the candidate competing strategies—decision variables of the corresponding agent. Each species is evolved through the repeated application of a conventional GA. Fig. 4 also shows the fitness evaluation phase of the GA. For example, to evaluate an individual from the security agent, that individual must collaborate with representatives of the other species (performance agents). Then, the system environment model solves for the system response. The security species can use the system response variable to evaluate the fitness of its individual. Here, the fitness function is the metric of the corresponding agent.

For the representative selection, there are many possible methods for choosing the representatives with which to collaborate. In order to facilitate the fast convergence of the evolutionary process, we use a “greedy” method for selecting representatives. In this method the current best individual from each species is selected as the representative of that species. A simple roulette wheel selection method is used to generate the reproduction operator of each species.

From the evaluation process above, we can see that the species are coordinated by the system environment response. When one agent changes its decision variables to gain a better fitness value, it will change the system response according to system dynamics and, in turn, change the fitness values of the other agents. Other agents will behave in the same way. The adjustment process will continue until no agent can gain better fitness value by changing its own decision variables without changes of the decision variables of other agents. In other words, the tradeoff finally reaches an optimal Nash equilibrium [29].

VI. SIMULATION AND RESULTS ANALYSIS

To obtain the system’s optimal performance and security tradeoff and validate the algorithms described in this paper, we developed a Simulink-based test-bed—iSpace Simulator—to evaluate the tradeoff model in iSpace in real-time with utility constraints and varying inputs.

A. Test-Bed Description

The iSpace simulator is made up of the following parts:
- Lego Mobile Robot: The differential drive mobile robot used in this platform has been constructed out of modular off-the-shelf LEGO Mindstorms NXT, which provides an easy-to-use set of pieces and convenient motors with encoders. The robot has a wheel radius of 0.0228 meters (m) and a distance between wheels of 0.095 m. With the gearing of the NXT motors, the maximum speed for this UGV is approximately 0.4 meters per second (m/s).
- Networked Supervisory Controller: The supervisory controller is designed to combine the distributed sensor information and make control decisions to accomplish the system goal. It is implemented in a host PC. In addition to implementing the data fusion and connecting distributed sensors, the supervisory controller also makes control decisions (e.g., the QC path-tracking algorithm).
- Security Simulator: Different security algorithms (e.g., DES, AES) were implemented on the security simulator developed in Labview in order to study the effects of various security mechanisms. AES is only considered in this paper. Thus, \( N_i \in \{128, 192, 256\} \).
- Network Communication Simulator: The main control and feedback information flow between the robot and the base station is carried over Bluetooth with supplementary sensing data (e.g., coordinates of the UGVs) coming...
• **Graphical User Interface (GUI):** To provide a user interface for the system, the GUI communicates directly to the base station (supervisory controller). Through the GUI, the user can control connections to the UGVs. The user can issue manual commands to each of these connections and display the data acquired from them as well. The progress and status of the UGVs are updated via the GUI while the UGVs are tracking the paths. Finally, the GUI also enables manual control and tuning of the UGVs through a command interface.

The test-bed lets us design repeatable experiments under a wide range of network environments, system conditions, and network settings. The system structure of the entire test-bed is depicted in Fig. 5.

### B. Simulation Conditions

Four UGVs are used in the simulation, so four test paths as shown in Fig. 6 are used for the four UGVs that represent them in order to test the implemented method. The UGVs will be commanded to follow test paths using average bandwidth allocation and CGA-based bandwidth allocation with varying network delays, as characterized in Fig. 6. We compare the effects of roundtrip time delays of 60, 200, 400, and 600 ms.

In order to facilitate testing, a software-based delay generator was also implemented. This delay generator acts on control signals and feedback to replicate the effects of network delay. To characterize the delay in this system, the UGV was pinged repeatedly with different delay settings for the delay generator. With the delay generator turned to 200 ms, it will add 200 ms of delay to each direction of communications. The results shown in Fig. 7 indicate that when the delay generator is turned on at 60, 200, 400, and 600 ms, the mean delays were 73, 237, 445, and 647 ms, respectively.

The key variables and parameters of the CGA are defined in Table III. The operational bounds for each agent are shown in Table IV.

### C. Simulation Result (Different CGA Parameters)

To compute the optimal allocated bandwidth for each agent, we apply CGA optimization to four cases in order to draw statistically significant results. The CGA parameters for the four test cases are listed in Table V. The time delay is 60 ms.

For each case, we run the simulation repeatedly 50 times to get the average results. The simulation results are listed in Table VI. Although they are on different convergence curves, all cases reach this problem’s Nash equilibrium, which is \((0.026, 0.067, 0.016, 0.043, 1.06)\) with the corresponding decision variables \(\{\text{HW}_1 = 20.68\%, \text{HW}_2 = 55.46\%, \text{HW}_3 = 3.76\%, \text{BW}_4 = 14.1\%, \text{BW}_{\text{sec}} = 6\%\}\).

As we can see from the results in Table VI, the fitness values in all cases converged to the Nash equilibrium rapidly after a
TABLE III  
KEY VARIABLES AND PARAMETERS OF CGA

<table>
<thead>
<tr>
<th>Individual</th>
<th>A set of decision variables for one species.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population</td>
<td>All of the individuals in one species. Individuals of each species are randomly initialized within the constraints.</td>
</tr>
<tr>
<td>Generation</td>
<td>Number of iterations in the evolutionary process.</td>
</tr>
<tr>
<td>Reproduction</td>
<td>Simple roulette-wheel selection.</td>
</tr>
<tr>
<td>Crossover</td>
<td>Two-point crossover operator: Two crossover positions are selected, and the two parents swap the strings between them with certain probability.</td>
</tr>
<tr>
<td>Mutation</td>
<td>Bit-wise mutation operator: Each bit of the individual is randomly changing from 1 to 0, or vice versa, with certain probability.</td>
</tr>
<tr>
<td>Elist Strategy</td>
<td>Preserving the Elite Rate of the best individuals in the previous generation.</td>
</tr>
<tr>
<td>Representative Choice</td>
<td>Current best individual from each species.</td>
</tr>
<tr>
<td>Termination Condition</td>
<td>The number of generations required to reach the preset upper limit.</td>
</tr>
</tbody>
</table>

TABLE IV  
OPERATIONAL BOUNDS OF AGENTS

<table>
<thead>
<tr>
<th>Performance Agent</th>
<th>Lower Bound</th>
<th>Upper Bound</th>
<th>Security Agent</th>
<th>Lower Bound</th>
<th>Upper Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_1$</td>
<td>0.01</td>
<td>0.25</td>
<td>$S$</td>
<td>1</td>
<td>1.15</td>
</tr>
<tr>
<td>$P_2$</td>
<td>0.03</td>
<td>0.50</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$P_3$</td>
<td>0.01</td>
<td>0.20</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$P_4$</td>
<td>0.02</td>
<td>0.50</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TABLE V  
CGA PARAMETERS OF THE FOUR TEST CASES

<table>
<thead>
<tr>
<th>Test case #</th>
<th>population size</th>
<th>generations</th>
<th>mutation rate</th>
<th>crossover rate</th>
<th>elite rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
<td>500</td>
<td>0.4</td>
<td>0.6</td>
<td>0.1</td>
</tr>
<tr>
<td>2</td>
<td>100</td>
<td>500</td>
<td>0.3</td>
<td>0.6</td>
<td>0.1</td>
</tr>
<tr>
<td>3</td>
<td>100</td>
<td>500</td>
<td>0.2</td>
<td>0.7</td>
<td>0.1</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
<td>500</td>
<td>0.1</td>
<td>0.8</td>
<td>0.1</td>
</tr>
</tbody>
</table>

TABLE VI  
NASH EQUILIBRIUM RESULTS OF THE CGA PROCESS

<table>
<thead>
<tr>
<th>Test Case #</th>
<th>Performance Value</th>
<th>Security Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>UGV1</td>
<td>UGV2</td>
</tr>
<tr>
<td>1</td>
<td>0.0266</td>
<td>0.0668</td>
</tr>
<tr>
<td>2</td>
<td>0.0265</td>
<td>0.0666</td>
</tr>
<tr>
<td>3</td>
<td>0.0265</td>
<td>0.0669</td>
</tr>
<tr>
<td>4</td>
<td>0.0266</td>
<td>0.0668</td>
</tr>
</tbody>
</table>

certain number of generations of coevolution. Thus, the CGA has a relatively high efficiency in the tradeoff optimization problem. Fig. 8 shows the tracking result of all the UGVs from test case 2. We can see that the system with the optimal bandwidth allocation has a very good path-tracking performance for all four UGVs.

TABLE VII  
PERCENTAGE IMPROVEMENT IN PERFORMANCE AND SECURITY WHEN USING CGA OPTIMAL BANDWIDTH ALLOCATION

<table>
<thead>
<tr>
<th>Delay</th>
<th>Percentage Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Performance</td>
</tr>
<tr>
<td>60ms</td>
<td>85%</td>
</tr>
<tr>
<td>200ms</td>
<td>74.5%</td>
</tr>
<tr>
<td>400ms</td>
<td>50.9%</td>
</tr>
<tr>
<td>600ms</td>
<td>82.3%</td>
</tr>
</tbody>
</table>

D. Simulation Result (Different Time Delays)

To compare and evaluate the optimal bandwidth allocation calculated by the CGA ($BW_1 = 20.68\%$, $BW_2 = 55.46\%$, $BW_3 = 3.76\%$, $BW_4 = 14.1\%$, $BW_{sec} = 6\%$) in different network environments, we run several comparison simulations with the average bandwidth setting ($BW_{1-4} = 20\%$, $BW_{sec} = 20\%$) under different time delay settings.

Table VII shows the improvement in the performance and security values of the CGA’s optimal bandwidth allocation compared to the average bandwidth allocation at different time delays. This value was computed as follows:

$$\text{improvement} = \frac{P_{average} - P_{CGA}}{P_{average}} \times 100\%.$$  \hspace{1cm} (19)

For each delay setting, we run the experiments 10 times to get the average results. As we can see, the optimal bandwidth allocation using the CGA has an impressive improvement in performance metrics while it still maintains the same security level compared to the other settings. From the simulation results, we conclude that the proposed CGA paradigm provides a satisfactory modeling and optimization scheme for the multiagent performance and security tradeoff problem on D-NCS.

VII. CONCLUSIONS AND FUTURE WORK

This paper addresses and defines the performance and security tradeoff problem of D-NCS and proposes a tradeoff model for performance and security in the D-NCS, as well as a paradigm for multiagent tradeoff optimization based on the CGA.
Simulations show that the CGA paradigm provides satisfactory modeling and optimization results for the performance-security tradeoff on the iSpace system. Thus, the coevolutionary paradigm presented in this paper is an effective approach for performance-security tradeoff analysis and optimization on D-NCS.

Future work include how to improve the tradeoff model when other security mechanisms are involved, e.g., quality of service (QoS) and intrusion detections.

REFERENCES


Wente Zeng (S’09) received the B.S. and M.S. degree in automation from Shanghai Jiaotong University, Shanghai, China, in 2006 and 2009, respectively. He is currently working toward the Ph.D. degree in electrical engineering at the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh. He has been a part of the Advanced Diagnosis, Automation, and Control (ADAC) Laboratory, North Carolina State University, since September 2009. His current research interests include networked control systems, distributed control, and system security.

Mo-Yuen Chow (S’81–M’82–SM’93–F’07) received the B.S. degree from the University of Wisconsin, Madison, in 1982, and the M.Eng. and Ph.D. degrees from Cornell University, Ithaca, NY, in 1983 and 1987, respectively. Upon completion of the Ph.D. degree, he joined the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, and has held the rank of Professor since 1999. He has been applying his research to areas including mechatronics, power distribution systems, distributed generation, motors, and robotics. He has established the Advanced Diagnosis and Control Laboratory, NC State University. He has published one book, several book chapters, and over one hundred journal and conference articles. His research focuses on fault diagnosis and prognosis, distributed control, and computational intelligence. Dr. Chow is the Editor-in-Chief of the IEEE TRANSACTION ON INDUSTRIAL ELECTRONICS. He has received the IEEE Region-3 Joseph M. Biedenbach Outstanding Engineering Educator Award the IEEE ENCS Outstanding Engineering Educator Award.