Zhang et al., Research in Engineering Design, 19(2-3):113–125, 2008

Evolutionary engineering design synthesis of on-board traffic monitoring sensors

Yizhen Zhang · Erik K. Antonsson · Alcherio Martinoli

Abstract In this paper, a formal engineering design synthesis methodology based on evolutionary computation is presented, with special emphasis on the design and optimization of distributed independent systems. A case study concerned with design of a sensory system for traffic monitoring purposes is presented, along with simulations of traffic scenarios at several levels of abstraction. It is shown how the methodology introduced is able to deal with the engineering design challenges present in the case study, and effectively synthesize novel design solutions of good quality. Moreover, when the fitness function is formulated as an aggregation of design preferences with different weights and trade-off strategies, the complete Pareto optimal frontier can be determined by the evolutionary synthesis methodology. The results of this study suggest that the approach can be useful for designers to solve challenging engineering design synthesis problems.

Keywords Engineering design synthesis · Evolutionary computation · Multi-level simulation · Engineering trade-offs and performance evaluations · Traffic systems · Distributed sensory systems

Y. Zhang · E. K. Antonsson (✉)
Engineering Design Research Laboratory,
Division of Engineering and Applied Science,
California Institute of Technology,
1200 East California Blvd., Pasadena,
CA 91125-4400, USA
e-mail: erik@design.caltech.edu

A. Martinoli
Distributed Intelligent Systems and Algorithms Laboratory,
École Polytechnique Fédérale de Lausanne,
1015 Lausanne, Switzerland

1 Introduction

Engineering design has traditionally been a creative process that requires human ingenuity and experience. In a modern engineering development process, highly challenging design tasks such as the development of intelligent vehicles (Martinoli et al. 2002; Zhang et al. 2003a) are characterized by severe reliability and robustness requirements, where each unit consists of an intelligent vehicle and a human operator. The main challenges in designing such systems include, but are not limited to, the following difficulties: (1) high, or sometimes even a priori unknown, complexity of good design solutions;[1] (2) multiple objectives, competing factors, trade-offs and/or simultaneous hardware and software optimization requirements; (3) the evaluation process and result for a given design solution can be intrinsically dynamic and stochastic instead of static and deterministic (Zhang et al. 2003b). All these problems make it difficult for an engineer, using traditional engineering methods, to synthesize an appropriate design solution under challenging system design requirements such as a traffic system.

These types of engineering design problems are challenging in-part because of the complex relationship between values of design variables and performance values. Naturally, where the relationship is structured and smooth (perhaps even differentiable and convex), well-established optimization methods can readily identify regions of the design space where performance is acceptable or optimal. However, in the cases where the relationship is not well-behaved (e.g., highly non-linear,

[1] "Complexity" of design solutions here refers to a combination of the number of degrees of freedom and the amount of information required to fully describe the design.

non-convex, discontinuous, or non-homeomorphic), conventional optimization approaches can perform poorly. For novel engineering design problems, where neither the configuration(s) of viable designs (nor their complexity) are known a priori, the relationship between points in the configuration space and their corresponding performances is typically not well-behaved.[2] Stochastic techniques, such as genetic algorithms (GA), may be the only way to identify promising design configurations, because they can search over poorly behaved functions, and can adapt to variable complexity.

Natural evolution has been an inspiration for engineering design researchers to develop automatic design synthesis methods. Since the 1960's, there has been an increasing interest in simulating the natural evolution process to solve optimization problems, leading to the development of evolutionary computation (EC) methods (Bäck 1996; Goldberg 1989; Mitchell 1996), such as GA, evolution strategies (ES), evolutionary programming (EP), and genetic programming (GP), which can often outperform conventional optimization methods when applied to challenging real-world problems. The idea is to have a pool of candidate solutions evaluated in parallel, from which the "fittest" solutions are chosen to mate and breed new candidate solutions using stochastic operators. This procedure is iterated until the population converges or a preset condition is met.

Recent research has demonstrated the ability of EC methods to successfully synthesize novel design configurations in various engineering design application domains (Antonsson and Cagan 2001; Bentley 1999; Lipson and Pollack 2000; Nolfi and Floreano 2000). However, many conventional EC applications such as design parameter optimization have assumed a fixed design architecture with a single well-defined design objective, which is represented by a deterministic fitness function. Therefore, standard EC methods must be appropriately adjusted to deal with the current engineering design challenges mentioned above.

In this paper, an evolutionary computational synthesis methodology is introduced for designing and optimizing distributed independent systems[3] automatically (Martinoli et al. 2002; Zhang et al. 2003a, b; Antonsson et al. 2003). This methodology shows several characteristics that appear promising for the distributed independent system design challenges. First, it works off-line in realistic simulations, preventing the test of unsafe solutions directly on real hardware, yet realistic enough to be transported to real hardware. Second, it is platform-independent and system-oriented, i.e., it can be applied to different platforms each with respective special system constraints. Third, it can incorporate uncertainty in the problem efficiently and easily adapt to collective tasks. Finally, in comparison to traditional hand-coded design, the design solutions are automatically synthesized and the human engineering effort involved is minimized to the mathematical formulation of the desired performance and to the encoding of real problems in the search space of the stochastic exploration algorithm.

This evolutionary design synthesis methodology is applied to a case study problem, which is concerned with the configuration design of an on-board traffic monitoring sensory system for intelligent vehicles, addressing all the engineering design challenges enumerated above. To assist the engineers in the design decision-making process, this methodology is especially applied to generate the full family of Pareto optimal design solutions, representing the different engineering design trade-offs, under various conditions and formulations of the design problems.

The goal of this work is to demonstrate an approach to configuration synthesis, based on a stochastic search technique, that produces good performing designs. The case study problem chosen here (vehicle sensor configurations) is sufficiently challenging to demonstrate our design synthesis approach, yet simple enough so that the viability of the results can be assessed by inspection. The objective is to apply the method to problems of increasing difficulty, and thereby to demonstrate that good design solutions can be synthesized with this approach on problems beyond the capacity of intuition.

In the following sections, the evolutionary engineering design synthesis method is presented, including special features introduced to face the modern engineering design challenges. The case study problem is presented next, with encoding of a given sensory problem, the simulation tools employed, and the fitness function which aggregates the weighted design preferences, etc. Sample results obtained in the framework of this case study are then presented and discussed, including the approximate Pareto frontier evolved under different weights and trade-off strategies. The paper concludes with a brief discussion of future promising applications.

2 Design synthesis methodology

In this section, the automatic engineering design synthesis methodology is introduced in detail, its special features to handle the current engineering design challenges are described. Based on EC, this methodology is shown to be able to synthesize novel design configurations of good quality.

[2] Consider, for example, the dramatic impact of small changes in the shape of the structure of an automobile on the modes and frequencies of induced vibrations.

[3] For example, a collection of intelligent vehicles operated by human drivers. Our approach is equally applicable to distributed autonomous systems.

Based on GA and ES, the evolutionary optimization loop used is shown in Fig. 1. First, an initial population of solutions is generated randomly. Then, each individual is evaluated under a performance test for one evaluation span.[4] According to the evaluation results, i.e., the fitness of each individual, the *parent selection* scheme chooses pairs of parent solutions for crossover, promoting individuals with higher fitness. Crossover between the selected pairs of parents is conducted under certain crossover probability to generate pairs of offspring. Mutation is also applied to each gene of the original pool under certain mutation probability and generates more offspring. If the fitness is deterministic, then only the offspring (from both crossover and mutation) are evaluated, otherwise the original population may also be *re-evaluated* to get a better estimate of their true fitness values. The best individuals are then deterministically selected from both the original population and the offspring, i.e., elitist *generation selection*, to constitute the next generation. Hence an offspring will only replace an individual of the original population if it has a higher fitness, conforming to the $(\mu + \lambda)$-selection scheme in ES (Bäck 1996), which insures that the mean of the population fitness is generally non-decreasing[5] over generations. At the end of each generational loop the program verifies whether or not another generation is needed in order to meet a pre-established criterion for terminating the evolutionary run.

This evolutionary design synthesis methodology is especially developed to deal with the engineering design challenges described above. First, the encoding allows variable-length chromosomes, making it possible to evolve design solutions of suitable complexity (e.g., an appropriate number of design parameters) and optimize parameter values simultaneously. In this case, the initial pool will be generated to contain solutions of random complexity. The crossover and mutation operators have to be adjusted from the standard ones to conform to the variable-length chromosome encoding, which will be explained in detail below in Sect. 3.4.

Second, multiple competing design objectives can be expressed as design *preferences* using fuzzy sets (Otto and Antonsson 1991; Scott and Antonsson 1998). Each value of a design objective or performance indicator is assigned a preference value between zero (totally unacceptable) and one (completely acceptable). Each objective may have a different level of importance, or *weight*. The weighted preferences can be aggregated, with a certain degree of compensation *s*, to get the *overall preference*, which serves as the *fitness function* in the evolutionary methodology. The whole family of achievable engineering trade-offs can be evolved by varying the compensation and weight parameters. Simultaneous hardware and software optimization could also be addressed by co-evolution of the system morphology and controller (Lipson and Pollack 2000; Bugajska and Schultz 2000), which appears to be more promising than evolving the morphology or controller alone, but is not considered in this paper.

Third, when the evaluation process and result are non-deterministic, i.e., dynamic and stochastic, as characterized by real traffic scenarios investigated in the case study, multiple re-evaluations of the surviving individuals may be introduced when appropriate. In such cases, solutions are selected based not only on their one-time performance but

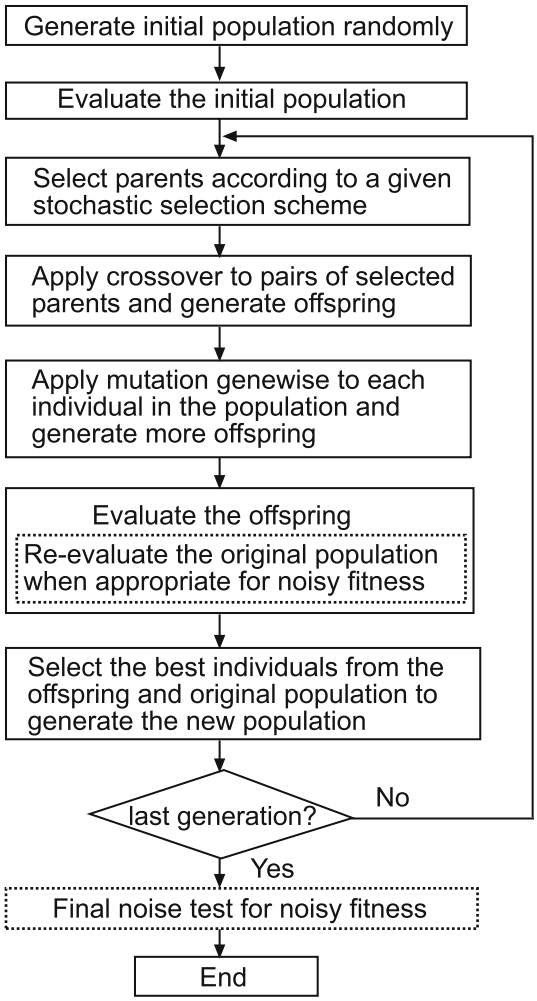

Fig. 1 The evolutionary optimization loop used in the automatic engineering design synthesis process

[4] When the abstracted tests (static, quasi-static or full coverage) are used, each evaluation span consists of performing a *coverage* test (described in Sect. 3.2) for all object vehicles placed into the detection region. When the embodied simulation is used, each evaluation span consists of 2,000 simulation time steps, corresponding to 128 seconds in real time.

[5] For the case of a non-deterministic fitness function, the individual fitness might decrease after re-evaluation, which in turn might cause the mean of the population fitness to decrease in some rare cases.

on an aggregation of multiple evaluation results (Zhang et al. 2003b), where the worst result could be taken as the overall fitness value when robustness of solutions is emphasized. While using the *worst* performance of each design is certainly a harsh measure of robustness, it is a fair comparison of the ability of each design to perform over the full range of conditions it is expected to encounter in operation. However, multiple evaluations of each individual (in cases where the evaluations are stochastic) can increase the computational cost, and a trade-off must be performed between the accuracy of the fitness estimate and the computational time required for the evolution process. In general,[6] the more expensive the evaluation tests are (compared to the genetic operations), the smaller the number of evaluations that should be used (Fitzpatrick and Grefenstette 1988; Paenke et al. 2006).

Finally, when the evaluations are non-deterministic, a final noise test is conducted, wherein the same number of evaluations are performed on all distinct individuals in the final population. This final noise test is used to determine the best performing and the most robust design solution. The results presented here are based on fitness evaluations that utilize pre-computed probability density functions reflecting the probabilities of vehicles appearing nearby the test vehicle. An earlier study (Zhang et al. 2003b) presented results based on stochastic fitness evaluations that directly incorporated uncontrolled variations (noise).

2.1 Benefits

The evolutionary design synthesis methodology presented here provides several benefits when generating design configurations.

By the nature of evolving and selecting individual design configurations from a population of designs, various design configurations with performances similar to the best result are also explored. This provides the engineer with an understanding of the range of configurations and performances available.

The unbiased nature of a stochastic exploration of a design configuration space (guided only by the performance of the resulting designs) has the potential to develop unconventional and non-intuitive configurations. The introduction of expected levels of uncontrolled variations (noise) in the evaluation tests can synthesize design configurations that are robust to those variations.

A range of trade-offs among a large number of competing performance measures can readily be explored. The approach presented in this paper does not require the decomposition of a large problem into individually optimized parts. Rather, the full configuration space can be explored with a balance of all performance measures as specified in the fitness function.

[6] At least one study has shown a savings in computational cost when evolving individuals in a noisy environment (Pugh et al. 2005).

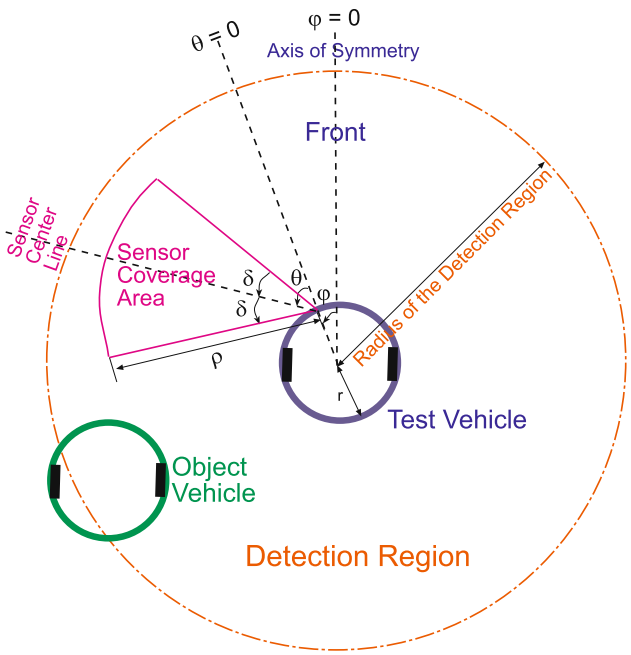

Fig. 2 Sensor parameters and the detection region: the sector shows a the scanning area of a sample sensor

3 Case study

The case study investigated in this paper is a simple problem in a dynamic and noisy environment. The goal is to determine the optimal configuration (such as number, type, and placement) of distance sensors on an intelligent vehicle, in order to monitor the traffic (i.e., other vehicles) in a pre-established detection region around the test vehicle in realistic traffic scenarios. The vehicle models considered here are circular with a single axle and two independent wheel-motors.[7] The detection region is also circular, as shown in Fig. 2. An object vehicle is considered detected by the sensory system if the body of the vehicle has overlap with at least one scanning area or ray of a sensor.

3.1 Encoding of sensor parameters

Sensors are mounted on the periphery of the vehicles, as shown in Fig. 2. The type and placement parameters, as well as the number of sensors, are the design variables to be determined and optimized according to the designer's

[7] Only minor alterations would be required to apply the method reported here to a conventional four-wheeled Ackerman-steering passenger vehicle.

and/or customer's preferences for values of various performance measures and the trade-off strategies chosen, which will be described in detail in Sect. 3.3. Except for the number of sensors which may take any positive integer values, all other design variables are encoded as discretized real numbers selected from predefined finite feasible ranges. The placement parameters of each sensor are characterized by two angles: the position angle φ (the angle between the front direction of the vehicle and the radius pointing to the position of the sensor) and the orientation angle θ (the angle between the radius pointing to the position of the sensor and the centerline of the scanning area of the sensor), as shown in Fig. 2. The type of each sensor is specified by its range ρ and cone of view δ. Therefore, each sensor is characterized by four design variables, and the number of design variables for a sensory system with n sensors is $4n$.

Each sensor also has a cost factor that depends on its range ρ and cone of view δ.[8] Typically the sensors with wider cones of view and longer ranges have a higher cost. This relationship can be determined from real sensor data or sensor models. A simple linear relationship is assumed in this case study:

$$\text{cost}_i = c_1\rho_i + c_2\delta_i + c_3 \tag{1}$$

$$\text{Total\_cost} = \sum_{i=1}^{n} \text{cost}_i \tag{2}$$

where cost_i, ρ_i, and δ_i are the cost, range, and cone of view, respectively, of the ith sensor; c_1, c_2, and c_3 are constant coefficients; n and Total_cost are, respectively, the number and the total cost of all sensors used in the current sensory system. Note that cost_i, ρ_i, and δ_i are all positive real numbers except that δ_i is also allowed to equal zero when the ith sensor is a line sensor.

As an important competing factor in the engineering design synthesis, the designer's and/or customer's preference for cost will be defined and incorporated into the fitness function introduced below in Sect. 3.3.

This seemingly simple case study problem reflects all of the engineering design challenges mentioned above. First, the optimal number of sensors is unknown, hence the number of design parameters as well as the complexity of the design solution is also open and increases with the number of sensors in the solution. Second, improving the coverage of the detection region and at the same time keeping a reasonable total cost for the whole sensory system are the two main design objectives here, whose relative importance lies in the aggregated fuzzy fitness function, described in Sect. 3.3, that leads to a trade-off between the two. Moreover, the evaluation process of candidate design solutions may be stochastic or deterministic, depending on the evaluation test used, which will be described in Sect. 3.2.

3.2 Evaluation tests

To understand the role of noise in shaping the evolved solutions, and to find out the best and most efficient evaluation tool for this case study, six different types of evaluation tests were implemented (Zhang et al. 2003b): static, one-dimensional (1D) and two-dimensional (2D) quasi-static, 1D and 2D full coverage, and an embodied test, as shown in Figs. 3 and 4. Static and full coverage tests are deterministic tests while quasi-static and embodied tests are stochastic tests, where different evaluation results (fitness values) are obtained by repeating the same evaluation test multiple times for a given solution.

As shown in Fig. 4, realistic sample traffic scenarios are simulated in an embodied simulator (Webots[9]), where a test vehicle and object vehicles are controlled by simple but realistic driver behaviors to move on a simulated three-lane highway. The vehicles either keep or change lanes to try to safely maintain their respective cruising speeds, and brake to avoid potential collisions. The sensors and actuators simulated are characterized by realistic noise values. Each vehicle is initialized with a random preferred cruising speed and an initial position for each evaluation span. As described above, candidate sensory configuration design solutions are mounted on the test vehicle to detect and monitor other object vehicles that enter the detection region of the test vehicle.

From the kinematic embodied traffic simulation described above, the relative distances and approaching angles of all other vehicles that have appeared around the test vehicle can be recorded at each time step and accumulated into the *vehicle occurrence data*. One-dimensional and 2D vehicle occurrence probability density functions (PDFs) can be generated from the normalized vehicle occurrence data, as shown in Figs. 5 and 6. Note that only the approaching angle of object vehicles is considered in the 1D PDF. These PDFs reflect the accumulation of vehicle occurrences for 5,000 evaluation spans, where each evaluation span contains 2,000 simulation time steps.[10] The number of evaluation spans used to generate the PDF is sufficient to capture the spatial distribution of vehicle occurrence probability, while averaging the temporal variations in traffic conditions.

[8] For a real sensor, in addition to its range and cone of view, the sensor cost may also depend on several other factors, such as accuracy, scanning frequency, power, sensory technology, etc., which are not included in this case study for simplicity.

[9] Refer to http://www.cyberbotics.com.

[10] Each evaluation span corresponds to 128 s in real time.

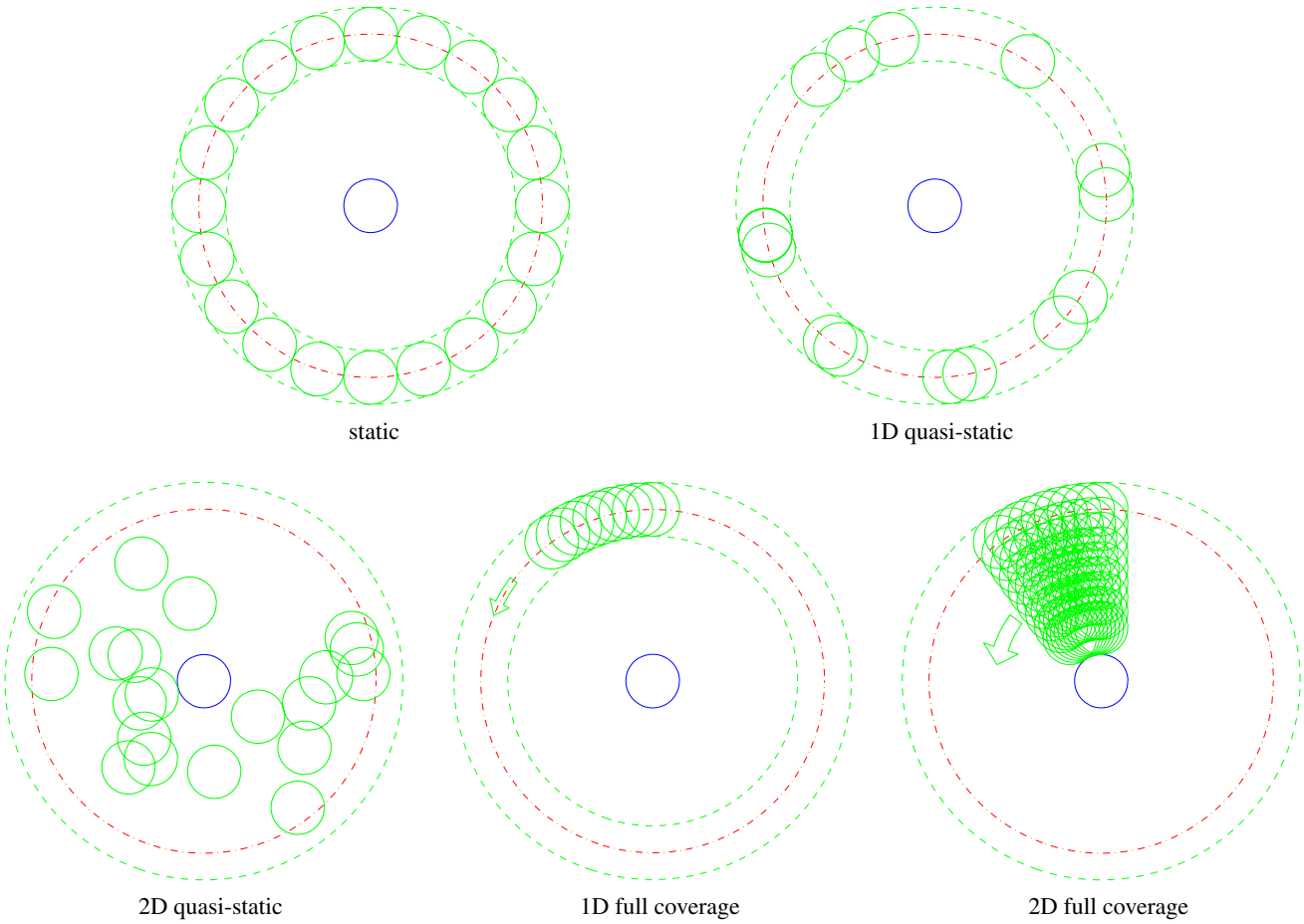

Fig. 3 Graphical representation of different types of evaluation tests

These PDFs capture the frequency of vehicle occurrence at each position within the detection region, which reflects the degree of priority to be covered by a sensory system in this case study. When complete coverage is not achievable, the regions with higher vehicle occurrence probability should get higher priority to be covered by the sensors.[11] Therefore, less computationally expensive and more abstracted traffic simulation tests, such as the quasi-static and full coverage tests, can be defined based on these PDFs.

In the quasi-static tests, as shown in Fig. 3, the test vehicle lies at the center statically, and object vehicles are placed randomly (according to the PDFs) on a ring (1D) or in the area (2D) within the detection region. While in the full coverage tests, the object vehicles are placed systematically along the ring (1D) or the area (2D) within the detection region, where the PDFs are used to weight the detection of the object vehicle at each position in order to estimate the coverage achieved by the current sensory solution in the traffic scenario represented by the PDFs, as explained in Sect. 3.3. In the static test, 20 static object vehicles are distributed evenly on the same 1D ring, representing a simple control experiment, which is not related to a realistic traffic scenario at all.

The six types of evaluation tests simulate the traffic scenarios with different levels of abstraction and significantly different simulation time costs, as shown in Table 1. In general, more realistic simulations are relatively more computationally expensive. To be more efficient, a new type of evaluation test could be a *hierarchical* test that combines some of these basic types of evaluation tests. In the hierarchical test, an abstracted simulation test is performed first as a pre-screening test prior to a more realistic and computationally more expensive test, which then in turn also serves as a pretest of an even more realistic and expensive test, and so on. Therefore, only if an individual solution performs well enough in a pretest, would it have a chance to be evaluated under a more realistic test higher in the hierarchy. In this way more computational time will be invested on more promising solutions, and poor solutions

[11] In addition to vehicle occurrences, historical accident statistics, as well as vehicle velocities, trajectories, and estimated threat level, might also be important factors to consider, but are not included in this case study.

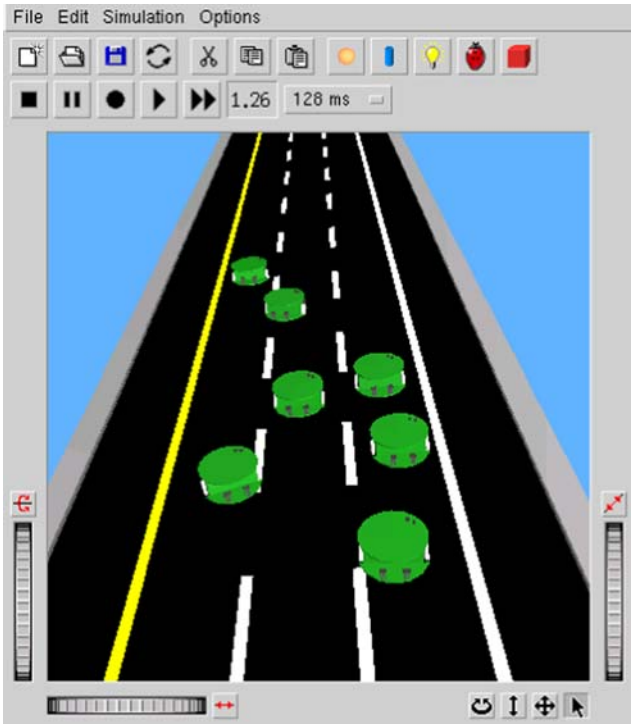

Fig. 4 Screen shot of the embodied simulator: Webots

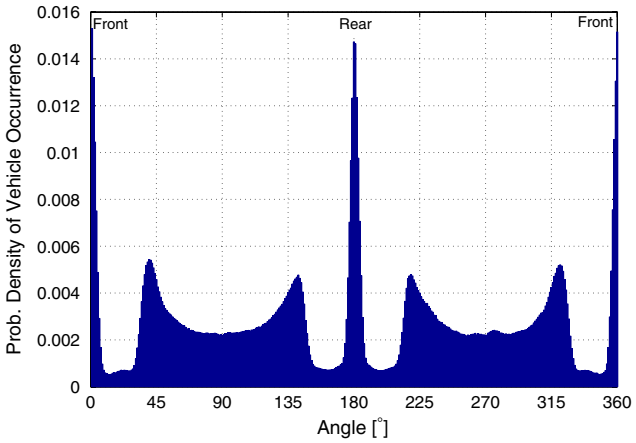

Fig. 5 1D PDF generated from the vehicle occurrence data collected in the embodied simulation for accumulative 5,000 evaluation spans

can be recognized and eliminated quickly with little simulation time cost. Although no results are presented in this paper based on a series of hierarchical tests, this approach will be applied in future work and is expected to provide additional computational savings, especially for cases where several evaluation tools of different levels of abstraction and computational costs are available.

It was shown in previous work (Zhang et al. 2003b) that evolutions under 2D full coverage and quasi-static tests can synthesize solutions of equivalent, if not better, quality than those under the embodied test. Based on this earlier

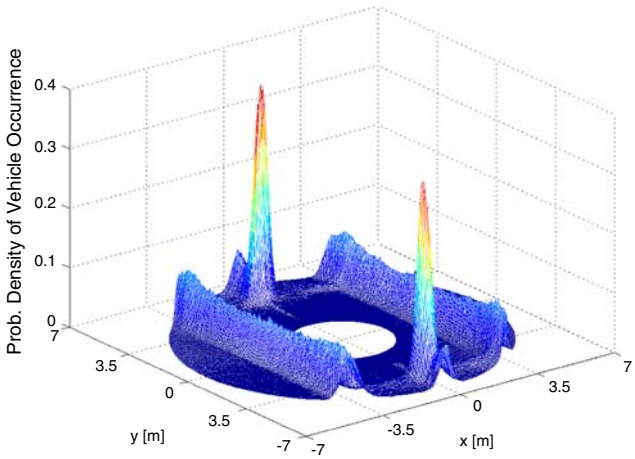

Fig. 6 2D PDF generated from the vehicle occurrence data collected in the embodied simulation for accumulative 5,000 evaluation spans: the test vehicle sits at the origin facing the positive *y*-axis

Table 1 Approximate relative time costs of different evaluation tests

Evaluation tests	Relative time cost
Static	1.0
1D quasi-static	3.4
2D quasi-static	116
1D full coverage	2.1
2D full coverage	134
Embodied: Webots	8,100

observation, results exclusively gathered with a 2D full coverage test are presented in this paper for simplicity and computational efficiency, since the 2D full coverage test is a *deterministic* implementation about 60 times faster than the embodied simulation, as shown in Table 1.

3.3 Fitness function

The objective of the example design problem presented here is to economically detect vehicles that may be on a collision course with the test vehicle with sufficient advance notice for the driver to avoid the collision. Therefore, the fitness function includes two elements: *coverage* and *cost*.

Coverage is the measure of the fraction of potentially colliding vehicles that are detected by a candidate sensor configuration. The objective of the design problem is to detect as many object vehicles as possible.

Cost is the measure of the projected cost of the candidate sensor configuration.

The balance of *coverage* and *cost* is the composite fitness function. The example problem statement does not make clear how important each measure is in this problem, nor does the problem statement indicate how much a good

value of one performance should compensate for a poor value of the other performance. To avoid making an ad hoc judgment of the proper balance, several different balances are illustrated here, followed by an exploration of all possible balances by producing the Pareto frontier.

The *coverage* under various evaluation tests is computed as follows:

$$\text{Coverage} = \sum_{i=1}^{V} k_i \cdot \text{PDF}(\alpha_i, r_i) \tag{3}$$

where V is the number of vehicles effectively appearing within the detection region during the evaluation span; k_i is one if the ith object vehicle is detected, or zero if it is not; α_i and r_i are the approaching angle and distance of the ith object vehicle relative to the test vehicle. The PDF is used to indicate the importance weighting of each particular position α_i and r_i (locations with high vehicle occurrence are more important than locations with low vehicle occurrence). For the full coverage tests, the PDFs used are generated from the vehicle occurrence data, as shown in Figs. 5 and 6. For all other tests, the PDF used in Eq. (3) is simply a constant ($1/V$) for any α_i or r_i and $V > 0$. Note that the probability of an object vehicle appearing in the test region at any [α_i, r_i] is governed by the PDFs shown in Figs. 5 and 6 for the quasi-static tests.

The fitness function is based on the overall preference for a candidate solution, which depends on the individual preferences given by the designer and/or the customer on all relevant performance criteria of the candidate solution. All design preferences here are expressed using fuzzy sets and take real values between zero (totally unacceptable) and one (completely acceptable).

In this case, the preference functions, μ_{coverage} and μ_{cost}, for the two competing factors, *coverage* and *cost*, showing the preferences for values of these performance measures, are simply given by:

$$\mu_{\text{coverage}} = (\text{Coverage})^2 \tag{4}$$

$$\mu_{\text{cost}} = \min\left\{1.0, \frac{20 - \text{Total\_cost}}{18}\right\} \tag{5}$$

as shown in Figs. 7 and 8. Equations (4) and (5) are *example* preference functions chosen to illustrate the method. In a real application of the method, these functions would reflect the design engineer's (or the customer's) actual preferences for *coverage* and *cost*. These simple curves are chosen for convenience in this case study, the same methodology can be applied with any preference functions.

Note that the values of *Coverage* always fall in the range [0, 1], reflecting the full range from no coverage to full coverage, while Total_cost is bounded below by zero. Hence the preference functions only need to be defined for these valid ranges.

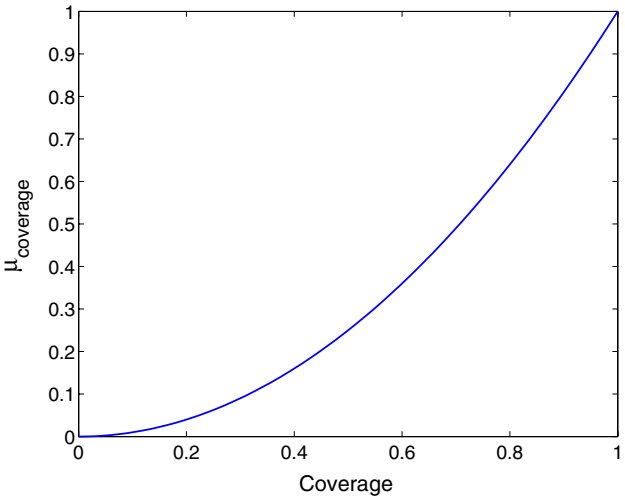

Fig. 7 Preferences for Coverage

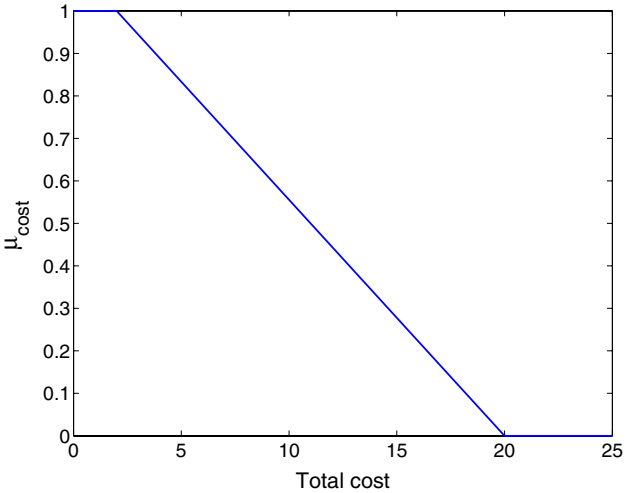

Fig. 8 Preferences for Total_cost

A common way to construct the multi-criteria fitness function is to assign importance weights to each criterion, and then aggregate the weighted preferences into an overall preference. The best design will have the highest overall preference. All current multi-criteria decision-making ultimately relies on the aggregation of disparate preferences with *aggregation functions*. The axioms that an aggregation function should obey to insure rational design decision-making were presented in Otto and Antonsson (1991). It was also shown (Scott and Antonsson 1998) that there is a family of aggregation function operators $\mathcal{P}_s$ that spans an entire range of possible operators between min and max, of which the set $\{\mathcal{P}_s | s \leq 0\}$ has included all operators that satisfy the design axioms. The class of functional equations (Aczél 1966), known as quasi-linear weighted means, is given by

$$\mathcal{P}_s(\mu_1, \mu_2; \omega_1, \omega_2) = \left(\frac{\omega_1 \mu_1^s + \omega_2 \mu_2^s}{\omega_1 + \omega_2} \right)^{\frac{1}{s}}. \quad (6)$$

Here, μ_1 and μ_2 are individual preferences on desired performance criteria. The parameter s establishes the *degree of compensation*, or the *trade-off strategy* adopted by the designer, reflecting the preferences of the customer. Higher values of s indicate a greater willingness to allow higher values of preference for the value of one variable to compensate for lower preferences for other variable values. The parameters ω_1 and ω_2 are importance weights, and their ratio $\omega = \frac{\omega_2}{\omega_1}$ is sufficient to characterize the relative importance of the two performance criteria. The definition above is only for two attributes, but can be extended to cases involving more than two criteria.

It was also shown (Scott and Antonsson 1998) that:

$$\mathcal{P}_{-\infty} = \lim_{s \to -\infty} \mathcal{P}_s = \min(\mu_1, \mu_2)$$
$$\mathcal{P}_0 = \lim_{s \to 0} \mathcal{P}_s = \left(\mu_1^{\omega_1} \mu_2^{\omega_2}\right)^{\frac{1}{\omega_1 + \omega_2}}$$
$$\mathcal{P}_1 = \lim_{s \to 1} \mathcal{P}_s = \frac{\omega_1 \mu_1 + \omega_2 \mu_2}{\omega_1 + \omega_2}$$
$$\mathcal{P}_{\infty} = \lim_{s \to +\infty} \mathcal{P}_s = \max(\mu_1, \mu_2).$$

Therefore the min operator $\mathcal{P}_{-\infty}$ indicates no compensation at all among various criteria, and the weighted geometric mean $\mathcal{P}_0$ represents the highest degree of compensation in design-appropriate (i.e., $s \leq 0$) aggregation functions. Note that the common weighted sum $\mathcal{P}_1$ is one instance (where $s = 1$) of this family of aggregation functions. However, the weighted sum, as well as the max operator (where $s = \infty$), turn out not to be appropriate aggregation functions for rational engineering design (Otto and Antonsson 1991; Scott and Antonsson 1998). It was also shown that any Pareto optimal point can be reached by a choice of some combination of the weight ratio ω and design-appropriate degree of compensation s.

Based on the above results, the fitness function used in this case study is defined to be the overall preference aggregated as a generalized weighted mean of the individual preferences, and is given by

$$\text{Fitness}(\omega, s) = \left(\frac{\mu_{\text{cost}}^s + \omega \mu_{\text{coverage}}^s}{1 + \omega} \right)^{\frac{1}{s}} \quad (7)$$

where

$$\omega \equiv \frac{\omega_{\text{coverage}}}{\omega_{\text{cost}}} > 0 \text{ and } s \in [-\infty, 0].$$

The design goal here is to maximize the fitness of the detector configurations by maximizing both design preferences, which, according to the preference curves shown in Figs. 7 and 8, is equivalent to increasing the coverage of the detection zone while at the same time reducing the total cost of sensors. To get better coverage of the detection region, more sensors with wider cones of view and/or longer ranges are needed, which tends to increase the total cost of the sensing system. A trade-off has to be made between the two, and a key point is to choose the weight ratio ω and degree of compensation s that lead to a desirable trade-off between the coverage and system cost under the specific design requirements. Therefore it is important not to arbitrarily limit the range of Pareto optimal points that can be selected by choosing a predetermined trade-off strategy. A method for establishing ω and s for a given problem was presented in Scott and Antonsson (2000).

Different Pareto optimal solutions for this case study can be easily obtained by setting the weight ratio ω and degree of compensation s in Eq. (7) and letting the evolutionary algorithm automatically synthesize solutions according to each fitness function (Antonsson et al. 2003). From these results, the design engineer can ascertain what level of performance can be achieved under certain preference settings, along with the corresponding cost of the sensing system, even in an early stage of design. This will help guide the design decision to a desirable trade-off between the various competing design objectives.

3.4 Algorithmic parameters

A parent selection based on the roulette wheel scheme, an elitist generation selection, a one-point crossover, and a uniform mutation, are used in this case study. Due to the use of variable-length chromosomes, insertion and deletion are also introduced as mutation operators, in addition to standard mutations that change only gene values, to change the lengths of chromosomes, i.e., directly insert or remove a sensor module in this case. The one-point crossover had to be modified to ensure proper crossover operation between parents with chromosomes of different lengths, which is described below.

A simple and efficient solution (Lee and Antonsson 2000) is to identify each gene in a chromosome with an index value chosen from a preset range, from which the crossover point is randomly chosen and divides the index range into two subranges. Then the two parent chromosomes swap their genes of the same index subrange to generate two children. In this particular sensory configuration case study, each sensor can be considered to be a gene in the chromosome that encodes the sensory system. The crossover line can be randomly chosen along the position angle φ (as shown in Fig. 2) from its range $[0, 2\pi)$, i.e., randomly selecting a point along the periphery ring where the sensors are mounted and connecting with the vehicle center. Then the sensors of the parents between the crossover line and the zero line ($\varphi = 0$) are swapped in the

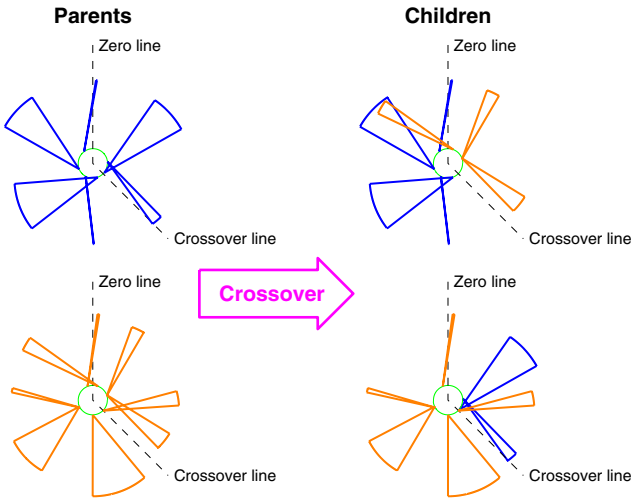

Fig. 9 Illustration of the one-point crossover scheme for two sensory systems with different numbers of sensors: the sectors (or lines) represent the sensor scanning areas (or rays)

crossover operation to generate two children, as shown in Fig. 9.

Table 2 summarizes the parameter values used in the evolutionary algorithm. The probabilities of genetic operators are fixed during an evolutionary run and are defined for each design solution.

4 Results

In this section, the automatic design synthesis method is applied to generate the best sensor configurations for the case study problem described above under different conditions, i.e., fitness functions with different values of the weight ratio ω and degree of compensation s, which reflect the relative importance between the two competing factors, *coverage* and *cost*, and how much higher preference values compensate for lower ones.

The evolutionary runs are conducted under the 2D full coverage evaluation test based on the 2D traffic PDF shown in Fig. 6. For simplicity, the sensor configurations are forced to have "modified" left-right symmetry[12] in the evolutions, conforming to the traffic PDF used. For each different experiment, evolutionary runs are repeated 10 times with different random number generator seeds and terminated after 200 generations for each run. Each initial population is randomly generated with sensory systems with a randomly chosen number of sensors from 2 to 20, and the final optimal number of sensors is determined by the evolutionary algorithm.

[12] The sensors lying near the symmetry axis itself are mirrored to the opposite end, as shown in Fig. 10.

Table 2 Algorithmic parameters

Parameters	Values
Population size	50
Selection scaling factor	2
$p_{\text{crossover}}$	0.2
p_{mutation}	0.182
$p_{\text{insertion}}$	0.05
p_{deletion}	0.05

Although it is not guaranteed that a global optimum from a strict mathematical point of view can always be generated, an evolutionary algorithm is able to discover some good and near-optimum solutions suitable for the engineering design use. Highly tuned systems are often sensitive to small imperfections, so engineers commonly design solutions to be slightly suboptimal to avoid such problems and increase robustness (Newman 2000).

Figure 10 shows the results obtained from evolutionary design synthesis experiments under three different conditions. The graphs in the upper row show the evolution processes of the mean *Fitness* of the population, as well as the two individual preferences, μ_{coverage} and μ_{cost}, over 200 generations; while the lower row shows the corresponding best sensor configurations evolved under each specific condition with their respective values of Coverage and Total_cost.

The left column of Fig. 10 shows the evolutionary result of an experiment with the weight ratio $\omega = \frac{3}{17}$ and the degree of compensation $s = 0$, which indicates that reducing cost is considered to be relatively more important than increasing coverage, and that the higher individual preference (μ_{cost}) can compensate for the lower one (μ_{coverage}). As a result, a simple and inexpensive sensory system of four sensors with low cost and low coverage is selected to cover only the regions with apparently high vehicle occurrence probability, as shown in Fig. 6.

On the other hand, the right column shows the result of an experiment with the weight ratio $\omega = 4$ and the degree of compensation $s = 0$, which means that the emphasis is on obtaining better coverage rather than reducing cost, and that the same trade-off strategy is adopted with opposite effects, i.e., the higher individual preference (μ_{coverage}) can compensate for the lower one (μ_{cost}). Consequently, a rather complex and expensive sensory system with eight sensors is evolved to cover most areas of the detection region with a coverage estimate of 98%.

Finally, the middle column of Fig. 10 shows the result of a special case with the degree of compensation $s = -\infty$, which means that the minimum of the two individual preferences is taken to be the overall preference regardless of their relative weights, i.e., a non-compensating trade-off

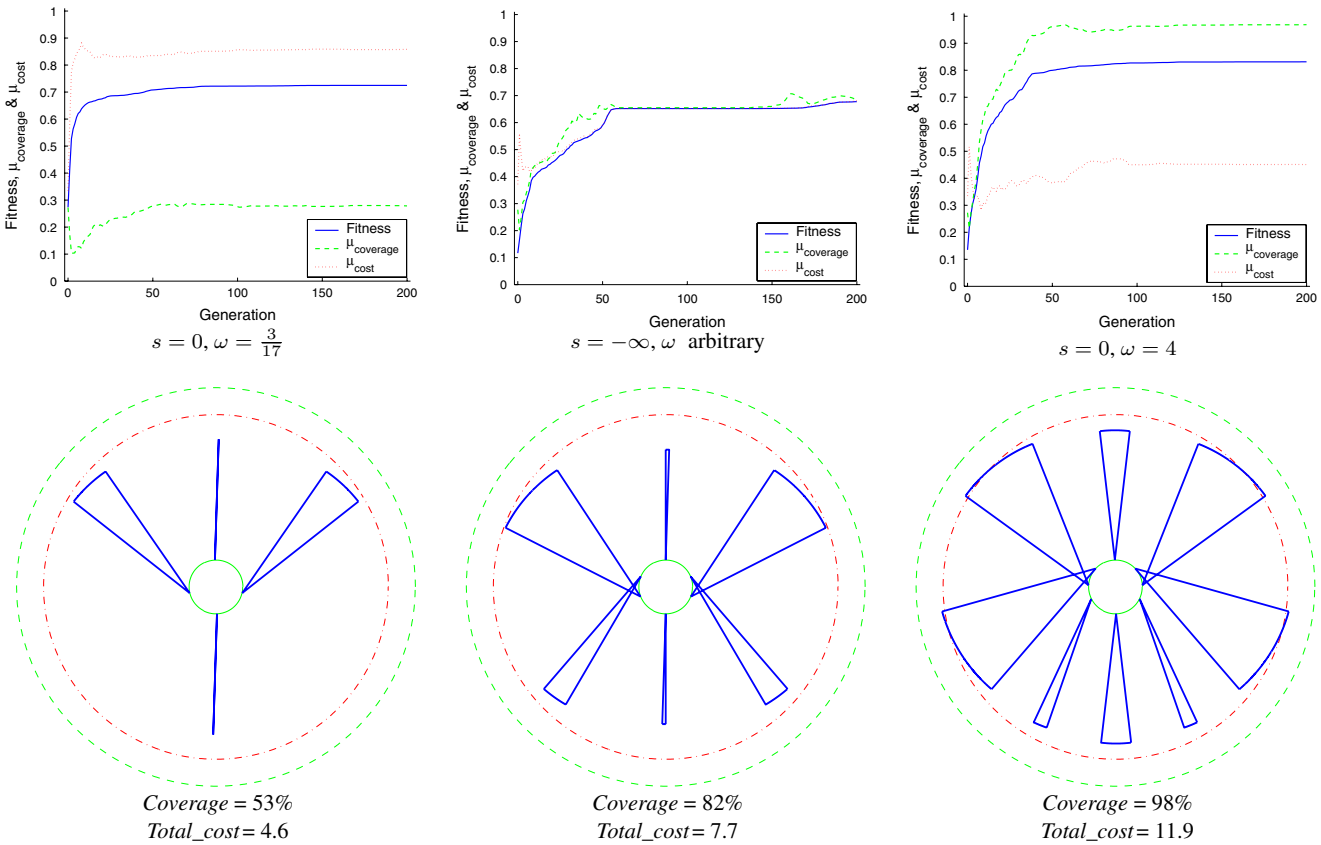

Fig. 10 Evolution of the population mean fitness and preferences (*top*) and the best sensor configurations (*bottom*) evolved under different conditions

strategy is adopted. A sensory system of medium cost and coverage is selected by the evolutionary algorithm in this case.

Notice that in all three configurations shown in Fig. 10 the forward-looking sensors are positioned nearly as far aft on the vehicle as possible (consistent with visibility through the sensing angle). Conventional sensor configurations place the forward-looking sensors on the forward-most positions on the vehicle. The advantage of the configurations generated here is that a sensor with a fixed angle of regard will sweep out a larger area in front of the vehicle. While this may not be a significant change from the conventional solution, it demonstrates the unbiased nature of a stochastic exploration of a design configuration space (guided only by the performance of the resulting designs), and shows the potential of methods such as this to develop unconventional and non-intuitive configurations.

As expected, the evolutionary engineering design synthesis methodology selects considerably different design solutions under different choices of fitness function parameters. More experiments based on different combinations of ω and s are performed and the set of the final best trade-offs evolved by the algorithm constitutes an approximate feasible Pareto optimal frontier for this design problem. Figure 11 illustrates the Pareto frontier by plotting the values of Total_cost versus Coverage of the best sensory configurations evolved with different fitness function parameters. Figure 12 shows the same Pareto frontier in terms of the preferences: μ_{cost} versus μ_{coverage}. Each data point represents the best result of one particular evolutionary experiment under a given combination of ω and s.

Figure 11 quantitatively outlines the general trend of the achievable coverage at various levels of cost. The coverage increases as the cost increases, and the rate of coverage increase lessens when the coverage approaches its upper bound 1, which agrees with common sense. It is desirable to maximize the preferences for both performance measures, i.e., reach the utopia point at the upper-right corner of Fig. 12, which is, however, impossible to be achieved. Therefore a trade-off has to be quantitatively established with an appropriate ratio of the relative importance (ω) and degree of compensation (s) between the two performance measures.

These results can be helpful for engineers in the design decision-making process. With the automatic design synthesis method presented here, these results can be obtained with minimum engineering effort and a modest

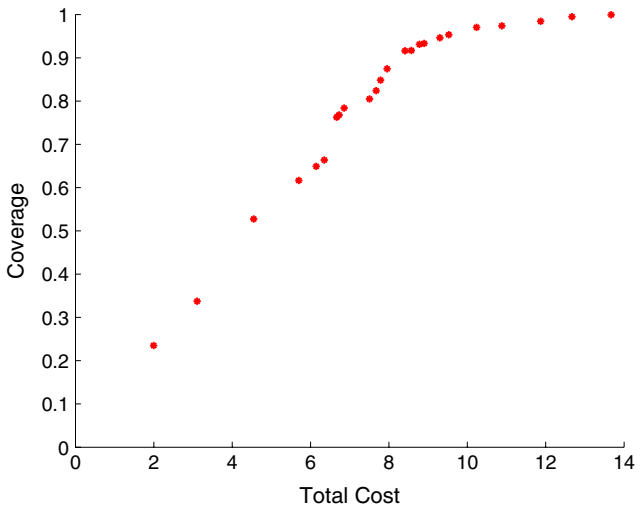

Fig. 11 Evolved Pareto frontier for the design trade-offs present in the case study: cost versus coverage

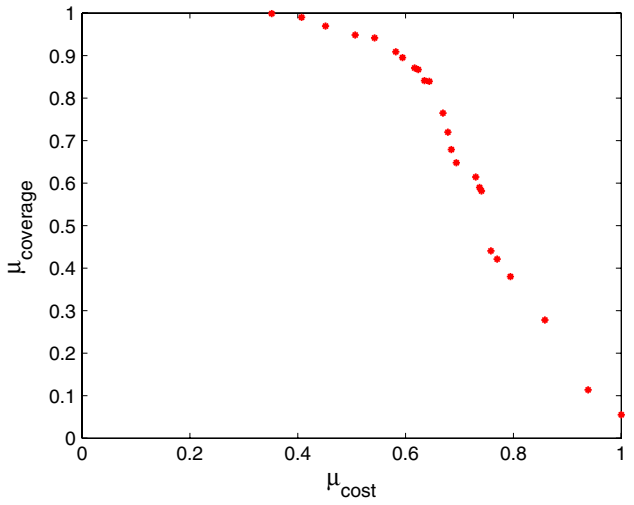

Fig. 12 Evolved Pareto frontier for the design trade-offs present in the case study, plotted in terms of preference for cost (μ_{cost}) and preference for coverage (μ_{coverage})

computational cost.[13] Although the best solutions found by the algorithm do not necessarily represent the optimal solutions under the specified conditions, they can quickly provide the design engineers with a general idea of various novel, promising configurations in the early stage of design.

Finally, the other parameters in the fitness function [Eqs. (1), (4), (5), (7)] and the evolutionary algorithm (Table 2) can also be adjusted by the design engineer to investigate their respective influences on the evolved results, to advance toward desired design goals.

[13] An evolution of 200 generations with the 2D full coverage test and a population of 50 individuals requires about 22 min of computational time on a computer with a 1.5 GHz AMD CPU.

5 Conclusion

An engineering design synthesis methodology based on evolutionary computation is presented, with special emphasis on dealing with the challenges present in design and optimization of distributed independent systems. This engineering design synthesis method is applied to generate different engineering design trade-offs utilizing fitness functions with different importance weighting ratios and trade-off strategies selected for multiple performance measures. Sample results of a case study concerned with the configuration problem of an on-board traffic monitoring sensory system are presented and discussed. The experimental results show that the proposed method can be efficiently applied to handle the engineering design challenges for distributed independent systems appropriately. Furthermore, the method is able to generate useful information to assist engineers in developing novel configurations and in the design decision-making process. Based on the results presented in this paper, this evolutionary engineering design synthesis method appears to be a promising approach for challenging engineering design synthesis problems.

Acknowledgments This material is based upon work supported, in part, by Delphi Delco Electronic Systems, and by the Engineering Research Centers Program of the National Science Foundation under Award Number EEC—9402726.

References

Aczél J (1966) Lectures on functional equations and their applications. Academic Press, New York

Antonsson EK, Cagan J (eds) (2001) Formal engineering design synthesis. Cambridge University Press, Cambridge

Antonsson EK, Zhang Y, Martinoli A (2003) Evolving engineering design tradeoffs. In: Proceedings of the fifteenth international conference on design theory and methodology (DTM), September 2003. ASME. Paper no. DETC2003/DTM-48676

Bäck T (1996) Evolutionary algorithms in theory and practice. Oxford University Press, New York

Bentley PJ (ed) (1999) Evolutionary design by computers. Morgan Kaufmann, London

Bugajska MD, Schultz AC (2000) Co-evolution of form and function in the design of autonomous agents: Micro air vehicle project. In: Proceedings of workshop on evolution of sensors GECCO 2000, Las Vegas, NV, August 2000, pp 240–244

Fitzpatrick JM, Grefenstette JJ (1988) Genetic algorithms in noisy environments. Mach Learn 3(2–3):101–120

Goldberg DE (1989) Genetic algorithms in search. Optimization and machine learning. Addison-Wesley, Reading

Lee C-Y, Antonsson EK (2000) Variable length genomes for evolutionary algorithms. In: GECCO 2000, proceedings of the genetic and evolutionary computation conference, July 2000. IEEE, Morgan Kaufmann, p 806

Lipson H, Pollack JB (2000) Automatic design and manufacture of robotic lifeforms. Nature 406:974–978

Martinoli A, Zhang Y, Prakash P, Antonsson EK, Olney RD (2002) Towards evolutionary design of intelligent transportation

systems. In: Fiorentin SR (ed) Proceedings of the eleventh international symposium of the associazione tecnica dell'automobile on advanced technologies for ADAS systems, Siena, Italy. ATA Paper Number 02A009

Mitchell M (1996) An introduction to genetic algorithms. MIT Press, Cambridge

Newman M (2000) Applied mathematics: the power of design. Nature 405:412–413

Nolfi S, Floreano D (2000) Evolutionary robotics. The MIT Press, Cambridge

Otto KN, Antonsson EK (1991) Trade-off strategies in engineering design. Res Eng Des 3(2):87–104

Paenke I, Branke J, Jin Y (2006) Efficient search for robust solutions by means of evolutionary algorithms and fitness approximation. IEEE Trans Evolut Comput 10(4):405–420

Pugh J, Zhang Y, Martinoli A (2005) Particle swarm optimization for unsupervised robotic learning. In: Proceedings of the swarm intelligence symposium, June 2005. IEEE, pp 92–99

Scott MJ, Antonsson EK (1998) Aggregation functions for engineering design trade-offs. Fuzzy Sets Syst 99(3):253–264

Scott MJ, Antonsson EK (2000) Using indifference points in engineering decisions. In: Proceedings of the eleventh international conference on design theory and methodology (DTM), September 2000. ASME

Zhang Y, Martinoli A, Antonsson EK, Olney RD (2003a) Evolution of sensory configurations for intelligent vehicles. In: Tsugawa S (ed) Proceedings of the IEEE intelligent vehicles symposium, Columbus, OH, June 2003. IEEE, pp 351–356

Zhang Y, Martinoli A, Antonsson EK (2003b) Evolutionary design of a collective sensory system. In: Lipson H, Antonsson EK, Koza JR (eds) Computational synthesis: from basic building blocks to high level functionality, Menlo Park, CA, 24–26 March 2003. Proceedings of the AAAI Spring Symposium Series, The American Association for Artificial Intelligence, AAAI Press, pp 283–290. Technical Report SS-03-02