Multi-Robot Box-Pushing in Presence of Measurement Noise

Pratyusha Rakshit, Amit Konar
Electronics & Telecommunication Engineering Department
Jadavpur University
Kolkata, India
pratyushar1@gmail.com, konaramit@yahoo.co.in

Atulya K. Nagar
Department of Mathematics and Computer Science
Liverpool Hope University
Liverpool, UK
nagara@hope.ac.uk

***Abstract*—Real-world multi-robot co-ordination problems, involving system (robot) design, control, and planning are often formulated in the settings of an optimization problem with a view to maximize system throughput/efficiency under the constraints on system resources. The paper aims at solving a multi-robot box-pushing problem in the presence of noisy sensory data using evolutionary algorithm. The process of co-ordination among multiple robots is characterized by a set of measurements and a set of estimators with a mathematical relationship between them (captured by the objective function). In the box-pushing problem by twin robots, the range data obtained by the robots at any instant of time are measurements, and the torque and/or force to be developed by the robots for a local movement of the box are estimators. We here use torques and forces developed by the robots to construct two objectives on minimization of energy consumed and time required for a local movement of the box in the presence of noisy sensory data. The box-pushing problem is solved using the proposed extended noisy non-dominated sorting bee colony (ENNSBC) algorithm to handle noise in the objective surfaces. Experiments undertaken in both simulation and real-world platforms reveal the superiority of the proposed ENNSBC to other competitor algorithms to solve the box-pushing problem with respect to the performance metrics defined in the literature.**

Keywords—multi-robot box-pushing; energy consumption; time requirement; measurement noise; multi-objective optimization.

I. Introduction

Since late 1980s, multi-agent co-ordination systems have emerged as a significant part of research in the realm of robotics. One of the vital challenges for multi-robot co-ordination systems is to design appropriate co-ordination strategies between the robots that enforce them to execute their task competently and time optimally in a complex workspace. Several works on multi-robot co-ordination have been reported in the literature [1−5]. Co-ordination typically has two alternative forms, *co-operation* and *competition*. When the success in one's goal causes a possible failure in the remaining agents' goal, we call it competition. In a co-operative scenario, agents usually have non-conflicting goals.

Box-pushing by twin robots is one of the most popular examples of multi-robot co-operation. The box-pushing problem aims at determining a continuous collision-free path for transportation of a box from a given starting position to a fixed goal position in an arbitrary rigid polyhedral environment by twin robots [1−2]. The transportation of the box can be determined globally or locally. The local planning is more flexible to its global counterpart because of its capability to take care of dynamic obstacles. Moreover, small time is required in local planning to identify the next position of the box only, rather than deriving the entire trajectory of motion for the box. Consequently, in this paper, the local planning is addressed to solve the box-pushing problem.

In the present context, we consider twin robots, capable to apply controlled torques and forces to jointly steer and translate a box respectively by a desired angle and distance. The box-pushing problem undertaken here aims at minimizing the total energy consumed and the total time required by the robots for the execution of the complete task. These two objectives are apparently conflicting. For instance, to reduce the total energy consumed for the transportation job, the twin robots have to apply less torques and forces, which in turn increases the time requirement. One modern approach to handle the conflicting objectives is to employ *multi-objective optimization* (MOO) techniques. The MOO technique to be used here will serve as a local path planner to determine the necessary torques/forces required along with the rotational/translational parameters of the box to move it locally with an objective of minimizing the energy consumed and time spent. Additionally, the merit of the paper lies in formulating the objective functions i) to confirm that the energy- and time-optimal local planning aligns the box towards its pre-defined goal position and ii) to ensure smoothness of the planned trajectory.

Traditional approaches of calculus-based MOO, usually, cannot be used to handle such optimization problem, primarily for two reasons. First, the objective functions occasionally are found to have multiple discontinuities distributed across the span of the variables. Second, the objective functions might change during real-time because of the dynamic nature of the measurement inputs. This calls for a derivative-free optimization technique. *Multi-objective evolutionary algorithms* (MOEAs) [6] suggest alternative approaches to handle such real-world MOO problems. The paper aims at formulating the multi-robot box-pushing problem in the settings of an MOEA, where the objectives of the MOEA include the fundamental goals of the co-operative robots (i.e., energy and time minimization to execute the complete task).

Significant research in robot co-ordination has been attempted by previous researchers in multi-agent robotics. However, unfortunately there are fewer or almost no traces of handling co-ordination problem in the presence of measurement noise in the sensory data. In the box-pushing problem, the range measurements taken by the sensors of the robots are often found contaminated with noise because of environmental constraints (causing path deviation due to multiple reflection of sonar/laser range signals/ or noisy sensor

characteristics). Trajectory planning of mobile robots evidently seems to be very difficult in the presence of noisy measurements.

In the present context, the energy- and time-objectives of MOEA being the functions of the sensory measurements of the robots of the box-pushing problem, an infiltration of noise in the measurement variables induces inaccuracies in the objective functions. Traditional MOEAs totally fail under such circumstances. The adverse effect of creeping of noise in the objective surfaces becomes prominent in the selection phase of an MOEA [6−10], [14]. The infiltration of noise in the objective function (often called fitness) estimates of a poor trial solution may deceive the selection operator and it may remain successful to get an accommodation in the population of the future evolutionary generation. Contrarily, a superior trial solution with its seemingly poor fitness estimates may be declined by the selection operator from being promoted to the next generation population. In this paper, we extend MOEAs under the settings of noisy objectives, and apply it in multi-robot box-pushing problem.

The optimization policy addressing the noise-induced uncertainty in the fitness assessments of a trial solution in an MOEA is referred to as *noisy MOEA* (NMOEA). The present problem of the multi-robot box-pushing in the presence of noise thus boils down to an intelligent MOEA problem, where interesting strategies need to be incorporated in a traditional MOEA to search optimal trial solutions in the presence of noise in the objective surfaces. In this paper, we have solved the multi-robot box-pushing problem using an extension of our previously proposed noisy non-dominated sorting bee colony (NNSBC) algorithm [9]. NNSBC is selected as the basic framework for NMOEA here to handle noise in the objective surfaces and for its simplicity in coding, fewer control parameters, good accuracy, and fast speed of convergence.

In NNSBC, the uncertainty in filtering quality trial solutions over evolutionary generations is handled by addressing three policies. First, the sample-size for periodic fitness (objective function) evaluation of each trial solution (to reduce the jeopardizing effect of noise to promote inferior solutions) is adapted with the fitness variance in its local neighborhood to proficiently balance the accuracy in fitness estimation and run-time. Second, the effective fitness of a trial solution is estimated from the expectation of its fitness samples, instead of conventional averaging. The third policy aims at including a slightly poor solution in the approximate Pareto front satisfying a statistical test.

In this paper, an extended version of NNSBC is proposed (referred to as *extended NNSBC*−ENNSBC henceforth). It differs from NNSBC in two aspects. First, the evaluation of the expected fitness of a trial solution of NNSBC is replaced by an alternative novel approach of the fitness expectation based on the distribution of its fitness samples in the entire fitness sample space. Here, a density-based nonuniform partitioning of the fitness sample space is employed to capture the uncertainty involved in the fitness measurement of the noisy fitness samples. Second, ENNSBC extends the deterministic Pareto dominance conditions of NNSBC by incorporating probabilistic estimate of dominance of a trial solution over other with an aim to identify true quality solutions.

Experiments are undertaken to study the relative performance of the proposed ENNSBC algorithm with respect to NNSBC [9] and other existing NMOEAs [6], [11], when energy- and time-objectives of the box-pushing problem are induced with measurement noise. Experimental results reveal that the proposed extension is capable of capturing better energy- and time-optimal paths for the transportation problem than those generated by the other competitive NMOEAs.

The paper is divided into seven sections. Section II provides a formulation of the multi-robot box-pushing problem. The issue of objective function selection is taken up in section III. Section IV introduces the NNSBC algorithm. In section V, we briefly outline the noise handling mechanisms in our proposed ENNSBC algorithm. Section VI reports the results of performance analysis of ENNSBC to solve the box-pushing problem. Conclusions are given in section VII.

II. Mathematical Formulation of the Box-Pushing Problem

In this section, the three steps involved in a single step of local movement of the box (Fig. 1) from the starting position to the next position are described. First, the robots turn the box, and then they translate it parallel to its length and lastly move it perpendicularly to its length. The box to be translated is considered to have a mass of M units. Its moment of inertia I about its centroidal z–axis perpendicular to the plane of the box is given by

$$I = M \times \left(l^2 + b^2\right) \big/ 12 \tag{1}$$

where l is the length and b is the breadth of the box.

A. Rotation About the Axis Passing Through the Center of the Box

Let $C(x_c, y_c)$ be the center of mass of the box and $E(x_e, y_e)$, $F(x_f, y_f)$, $G(x_g, y_g)$ and $H(x_h, y_h)$ be the four corners of the box respectively at some time t as shown in Fig. 1 (step 1). The expressions for the new position after rotation are given as

$$\left.\begin{aligned} x_i' &= x_c\left(1-\cos\alpha\right) + x_i\cos\alpha - \left(y_i - y_c\right)\sin\alpha \\ y_i' &= y_c\left(1-\cos\alpha\right) + y_i\cos\alpha + \left(x_i - x_c\right)\sin\alpha \end{aligned}\right\} \tag{2}$$

for all $i \in \{e, f, g, h\}$. Given the torque Γ, the energy E_1 required to rotate the box around an axis perpendicular to the X−Y plane, by angle α is

$$E_1 = \Gamma \times \alpha \tag{3}$$

and the time τ_1 required is obtained as

$$\tau_1 = \sqrt{\frac{2 \times \alpha \times I}{\Gamma}} \tag{4}$$

B. Translation Perpendicular to Width

In order to translate the box perpendicular to its width one robot pushes the box and the other pulls it. These forces move it by a distance r_w parallel to its length. As shown in the Fig. 1 (step 2), let the new co-ordinates of the corners the box after translation be $E''(x_e'', y_e'')$, $F''(x_f'', y_f'')$, $G''(x_g'', y_g'')$, $H''(x_h'', y_h'')$ and the center be $C''(x_c'', y_c'')$. The expression for the new co-ordinate of the vertices is given by the following equations.

$$\left.\begin{aligned} x_i'' &= x_i' + r_w\cos\alpha \\ y_i'' &= y_i' + r_w\sin\alpha \end{aligned}\right\} \tag{5}$$

for all $i \in \{c, e, f, g, h\}$.The energy E_2 consumed by the robots to translate the box by a given distance r_w is given by

$$E_2 = F_w \times r_w \tag{6}$$

where F_w is the sum of the pulling and pushing forces applied by the robots. The corresponding time τ_2 is obtained as,

$$\tau_2 = \sqrt{\frac{2 \times r_w \times M}{F_w}} \tag{7}$$

C. Translation Perpendicular to Length

In order to move the box perpendicular to its length, the robots pull the box in the same directions with forces parallel to its width. Let the sum of those forces applied be F_l and the new position of the vertices of the box be $E'''(x_e''', y_e''')$, $F'''(x_f''', y_f''')$, $G'''(x_g''', y_g''')$, $H'''(x_h''', y_h''')$ and the center be $C'''(x_c''', y_c''')$. Expression (8) gives the co-ordinates of the vertices after translation of the box.

$$\left.\begin{aligned} x_i''' &= x_i'' + r_l \sin\alpha \\ y_i''' &= y_i'' + r_l \cos\alpha \end{aligned}\right\} \tag{8}$$

for all $i \in \{c, e, f, g, h\}$. The energy E_3 required to bring about this movement is

$$E_3 = F_l \times r_l \tag{9}$$

and the time τ_3 required for translation is

$$\tau_3 = \sqrt{\frac{2 \times r_l \times M}{F_l}} \tag{10}$$

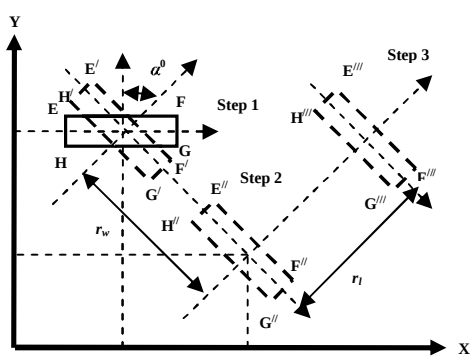

Fig. 1 The three steps involved in local planning: rotation of the box by α^0, and its translations along the length and the width by r_w and r_l respectively

III. Construction of the Objective Functions

In this paper, the energy consumed by the robots and the time required to execute the box-pushing task are considered as two conflicting primary objective functions [19−20]. The time and energy objectives need to be optimized here before each step of local movement of the box (for local planning) to select the optimum next position among many alternatives. The secondary objective (*so*) in the present context is concerned with i) the distance between the next position of the box and the goal position and ii) the smoothness of the traversed path. The secondary objective function in the present case ensures that the energy- and time-optimal optimization policy i) does not derive any new position moving away from the goal and ii) results in a smooth trajectory.

In the process of selecting next position of the box from its current position, we should take care that the next position is not in the close vicinity of obstacles/sidewalls of the robots workspace. This is ensured by a penalty function. The penalty function has a large value when the next position of the box is close enough to an obstacle or sidewall. It offers a small penalty when the next position is away from the obstacle or sidewall of the world map.

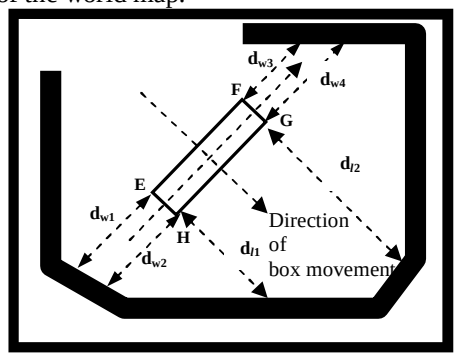

Fig. 2 Diagram illustrating the calculation of *d*

A. The Energy Objective

The first objective function considering the total energy required by the twin robots for one complete step of local movement is given by

$$J_1 = E_1 + E_2 + E_3 + so + penalty \tag{11}$$

where $so = K_e(|x_{gl} - x_{c,t}| + |y_{gl} - y_{c,t}|) + s(C_t)$

$$\text{and } s(C_t) = \max_{i=2}^{t} \frac{\phi_i}{\min(dist(C_{i-1}, C_i), dist(C_i, C_{i+1}))} \tag{12}$$

Here $C_t(x_{c,t}, y_{c,t})$, $C_{t+1}(x_{c,t}, y_{c,t})$ and $C_g(x_{gl}, y_{gl})$ respectively denote the positions of the center of mass of the box at the t-th (current) and the t+1-th (next) instants and at the pre-defined goal location and K_e is a constant. ϕ_i is the angle made by the extended line segment joining the trajectory points C_{i-1} and C_i with the line segment connecting points C_i and C_{i+1}. $dist(C_i, C_j)$ symbolizes the distance between the mass centers of the box at the i-th and j-th time instants. Minimization of $s(C_t)$ enhances the smoothness of the trajectory planned so far.

The penalty is defined as a function of distance d of the next position of the box from the obstacles and from the boundary wall of the workspace. It is given by

$$penalty = f_{st}/d \tag{13}$$

where f_{st} is a constant, and d is a function of distance of the box with obstacles and sidewalls, and is measured as

$$d = \min(d_{w1}, d_{w2}, d_{l1}, d_{l2}, d_{w3}, d_{w4}) \tag{14}$$

where d_{w1}, d_{w2}, d_{w3}, d_{w4}, d_{l1}, d_{l2} are the distances of the vertices of the box withfrom the static obstacles and the sidewall of the workspace as shown in Fig. 2. These are the range data obtained from the distance finding sensors of the robots. It is also seen that as the calculation of d depends on the noisy sensory data, the objective functions also become noisy.

B. The Time Objective

The time required for the twin robots to transfer the box to its next position in one-step is used to design the second objective function J_2, given by

$$J_2 = \tau_1 + \tau_2 + \tau_3 + so + penalty \tag{15}$$

where $so = K_t(|x_{gl} - x_{c,t}| + |y_{gl} - y_{c,t}|) + s(C_t)$

$$\text{and } s(C_t) = \max_{i=2}^{t} \frac{\phi_i}{\min(dist(C_{i-1}, C_i), dist(C_i, C_{i+1}))} \tag{16}$$

where K_t is a constant. Other symbols of (16) carry the same meaning as in J_1.

IV. An Overview of Noisy Non-dominated Sorting Bee Colony (NNSBC)

NNSBC, proposed in [9], is a population-based metaheuristic algorithm, capable to handle noise in the objective surfaces (fitness landscapes). An overview of NNSBC for minimizing N noisy objective functions is given below.

A. Initialization

NNSBC commences from a population $\mathbf{P}(G)=[\vec{Y}_1(G),\vec{Y}_2(G),...,\vec{Y}_{NP}(G)]^T$ of NP, D-dimensional food sources (representing trial/candidate solutions of the MOO problem) $\vec{Y}_i(G)=\{y_{i,1}(G),y_{i,2}(G),...,y_{i,D}(G)\}$ at the current generation G = 0 by uniformly randomizing in the range $\vec{Y}^{\min}=\{y_1^{\min},y_2^{\min},...,y_D^{\min}\}$ and $\vec{Y}^{\max}=\{y_1^{\max},y_2^{\max},...,y_D^{\max}\}$, given by

$$y_{i,j}(G)=y_j^{\min}+\text{rand}(0,1)\times(y_j^{\max}-y_j^{\min}) \tag{17}$$

for j=[1, D] and i=[1, NP]. Here rand(0, 1) is a uniformly distributed random number in [0, 1]. The sample size $n_{k,i}$ for the periodic evaluation of the k-th fitness $J_k(\vec{Y}_i(G))$ is set to the minimum sample size $n^{\min}$ for k=[1, N] and i=[1, NP].

B. Evaluation of Expected Fitness

First $J_k(\vec{Y}_i(G))$ is evaluated for $n_{k,i}$ times. Then the minimum and the maximum values of the observed fitness samples are recorded as $J_{k,i}^{\min}$ and $J_{k,i}^{\max}$ respectively. Now the entire range [$J_{k,i}^{\min}$, $J_{k,i}^{\max}$] is divided into pre-defined L intervals of equal length. The expected fitness $E_{k,i}$ and the variance $\delta_{k,i}$ are evaluated as

$$E_{k,i}=\sum_{l=1}^{L}p_l\times((l^{\min}+l^{\max})/2) \tag{18}$$

and $$\delta_{k,i}=\sum_{l=1}^{L}p_l\times((l^{\min}+l^{\max})/2)^2-E_{k,i}^2 \tag{19}$$

where p_l denotes the probability of occurrence of the fitness samples in the l-th interval (with boundary [$l^{\min}$, $l^{\max}$]), given by

$$p_l=n_l/n_{k,i} \tag{20}$$

with n_l representing the number of fitness samples in the l-th interval for l=[1, L]. This step is repeated for k=[1, N] and i=[1, NP].

C. Employed Bee Phase

An employed bee discovers a new food source $\vec{Y}_i'(G)=\{y_{i,1}(G),...,y'_{i,j}(G),...,y_{i,D}(G)\}$ in the neighborhood of $\vec{Y}_i(G)$ by computing $y_{i,j}'(G)$ using

$$y'_{i,j}(G)=y_{i,j}(G)+\text{rand}(-1,1)\times(y_{i,j}(G)-y_{k,j}(G)) \tag{21}$$

where rand(−1, 1) is a uniform random variable in [−1, 1] and j and k are randomly selected integers from [1,D] and [1, NP] respectively but $k\neq i$. This is repeated for i=[1, NP].

D. Sample Size Adaptation

The local neighborhood of $\vec{Y}_i'(G)$ is formed by a sub-population of food sources residing within a hyperspace bounded by$\{y_{i,1}'-\Delta y_1,\ y_{i,2}'-\Delta y_2,\ldots,\ y_{i,D}'-\Delta y_D\}$ and $\{y_{i,1}'+\Delta y_1,\ y_{i,2}'+\Delta y_2,\ldots,\ y_{i,D}'+\Delta y_D\}$ with $\Delta y_j=(y_j^{\max}-y_j^{\min})/NP$ for j= [1, D]. Once the neighborhood is identified, the sample-size, $n_{k,i}'$ is identified from the fitness variance $v_{k,i}$ in local neighborhood, given by

$$n'_{k,i}=\frac{n^{\min}+n^{\max}}{2}+\frac{n^{\max}-n^{\min}}{\pi}\times\arctan(v_{k,i}-Th_k) \tag{22}$$

where $n^{\max}$ represents the maximum sample-size and Th_k denotes the threshold for the k-th fitness variance. It is set equal to the lower quartile of the fitness variances in the neighborhood of each population member. After determining $n_{k,i}'$, $E_{k,i}'$ and $\delta_{k,i}'$ are evaluated following the step B. This is repeated for k=[1, N] and i=[1, NP]. This adaptive selection of sample size effectively balances the computational accuracy and the run-time complexity.

E. Dominance based Selection by Employed Bee

The new food source $\vec{Y}_i'(G)$ replaces $\vec{Y}_i(G)$ if $\vec{Y}_i'(G)$ dominates $\vec{Y}_i(G)$ [12]. If they are non-dominated, $\vec{Y}_i'(G)$ is appended to $\mathbf{P}(G)$. Repeating the step for i= [1, NP] yields a population of size $S\in[NP, 2NP]$.

F. Non-dominated Sorting and Pareto Co-ranking

The population $\mathbf{P}(G)$ is then sorted into a number of Pareto fronts (FS(1), FS(2), FS(3), and so on) according to non-domination [12]. After that, a slightly inferior food source $\vec{Y}_j(G)\in FS(f)$ with f >1, is included in FS(1) if

$$|E_{k,i}-E_{k,j}|<K\sqrt{\gamma_{k,i,j}\left(\frac{1}{n_{k,i}}+\frac{1}{n_{k,j}}\right)} \tag{23}$$

for k= [1, N]. Here $\vec{Y}_i(G)\in FS(1)$ and the pooled variance

$$\gamma_{k,i,j}=\frac{(n_{k,i}-1)\times\delta_{k,i}+(n_{k,j}-1)\times\delta_{k,j}}{n_{k,i}+n_{k,j}-2} \tag{24}$$

Here K is the neighborhood restriction factor [13]. The strategy hinders the loss of information (in noisy environment) by providing accommodation to a seemingly inferior solution in the approximate Pareto front FS(1), satisfying (23).

G. Truncation of the Extended Population

The parent population $\mathbf{P}'(G)$ (of size $NP<S$) for the onlooker bee phase is formed by selecting the non-dominated sets of solutions from $\mathbf{P}(G)$ (of size S) according to the ascending order of their Pareto ranking. When a Pareto front $FS(l)$ is found, which can be partially promoted $\mathbf{P}'(G)$, its members are again sorted in descending order of crowding distance CD [12]. The members with high CD measure are prioritized to be promoted to $\mathbf{P}'(G)$ until its size becomes NP.

H. Probability Calculation

The probability of each food source $\vec{Y}_i(G)$ to be selected by the onlooker bee is given by

$$prob(i)=|Set_i|/NP \tag{25}$$

where $|Set_i|$ symbolizes the number of all food sources being dominated by $\vec{Y}_i(G)$ for i=[1, NP].

I. Onlooker Bee Phase

An onlooker bee selects a food source based on its selection probability (as given by (25)) ad discovers a new member following (21). The sample size and the fitness estimates of the new member are evaluated following steps D and B. The population $\mathbf{P}'(G)$ is updated by following the principles as stated in steps E and F. The population $\mathbf{P}(G+1)$ for the next generation (of size NP) is then formed using the methodology of non-dominated sorting and crowding distance metric.

J. Scout Bee Phase

If the position of a population member cannot be enhanced after a pre-defined number of evolutionary generations called 'limit', it is replaced by a randomly reinitialized position by the scout.

After each evolution, we repeat from step C until termination condition for convergence is satisfied.

V. Extended Noisy Non-dominated Sorting Bee Colony (ENNSBC)

The NNSBC algorithm is extended using the following two proposed concepts by modifying its two steps: i) evaluation of the expected fitness and ii) the dominance-based food source selection by employed/onlooker bees, as introduced in section IV. The expected fitness evaluation phase is amended by considering the non-uniform quantization of the fitness sample space of individual trial solutions. The selection phase is extended by replacing the deterministic dominance criteria with a probabilistic estimate in the noisy environment.

The extended NNSBC, called ENNSBC is similar with NNSBC except modifications in steps IV.B and IV.E as given below.

A. Sample-Distribution-based Fitness Estimation

The alternative approach proposed to estimate the fitness of a food source in the noisy environment aims at partitioning the fitness sample space based on the density of its fitness samples. Then the expected value of the fitness samples is regarded as its true fitness estimate. The proposed strategy is concerned with biasing the true fitness estimate of a trial solution towards the fitness samples in the crowded zones in the sample space, while dealing with the rare fitness samples with less significance. It presumes that the rare fitness samples stem from the jeopardizing effect of noise. *Sample-distribution-based fitness estimation* (SDFE) includes four main steps:

(a) Selection of non-uniform intervals in the fitness sample space: We first determine the variance $V_{k,i}$ of the measured samples of the k-th fitness of trial solution $\vec{Y}_i(G)$, the minimum and the maximum values of the observed fitness samples $J_{k,i}^{\min}$ and $J_{k,i}^{\max}$ respectively. Now the entire range $[J_{k,i}^{\min},\ J_{k,i}^{\max}]$ is first partitioned into two intervals of equal lengths, respectively represented by $[J_{k,i}^{\min},\ J_{k,i}^{\mathrm{mid}}]$ and $[J_{k,i}^{\mathrm{mid}},\ J_{k,i}^{\max}]$, where

$$J_{k,i}^{\mathrm{mid}} = (J_{k,i}^{\min}) + J_{k,i}^{\max}) / 2 . \tag{26}$$

If the variance of the fitness samples lying in the first interval is found to be greater than $V_{k,i}/n_{k,i}$, it is again partitioned into two more equal-length intervals, represented by $[J_{k,i}^{\min},\ J_{k,i}^{\mathrm{mid},1}]$ and $[J_{k,i}^{\mathrm{mid},1},\ J_{k,i}^{\mathrm{mid}}]$, respectively, where

$$J_{k,i}^{\mathrm{mid},1} = (J_{k,i}^{\min} + J_{k,i}^{\mathrm{mid}}) / 2 . \tag{27}$$

The same policy is applied for the second interval also. The same approach is repeatedly adopted for all subsequent intervals until the variance of the fitness samples in each interval becomes less than $V_{k,i}/n_{k,i}$. Consequently, the entire sample space $[J_{k,i}^{\min},\ J_{k,i}^{\max}]$ is now divided into L intervals of unequal length as indicated in Fig. 3.

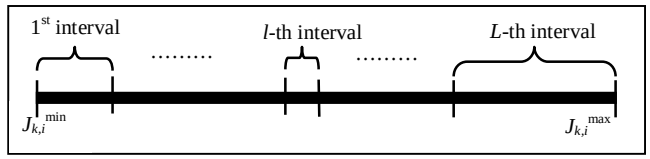

Fig. 3 Non-uniform fitness intervals in the sample space

(b) Probability of occurrence of the fitness samples: The proposed nonuniform partitioning of the fitness sample space indicates that the fitness samples in the lengthy intervals are rare samples, probably resulting from noise contamination. Therefore, their involvement in the true fitness estimation of $\vec{Y}_i(G)$ should not be considered similarly to that of the fitness samples in the shorter intervals. The contribution of the fitness samples of the interval l towards the fitness estimate of the trial solution $\vec{Y}_i(G)$ is quantitatively evaluated by the probability measure p_l as in (20) for l=[1, L].

(c) Expected fitness estimation: The expected estimate $\overline{J}_{k,i}$ of fitness $J_k(\vec{Y}_i(G))$ is obtained by

$$\overline{J}_{k,i} = \sum_{l=1}^{L} p_l \times J_{k,i}^{l} . \tag{28}$$

where $J_{k,i}^{l}$ denotes the median value of the fitness samples of $J_k(\vec{Y}_i(G))$ in the l-th interval, for l = [1, L]. The expected value thus obtained offers a distinct estimate of $J_k(\vec{Y}_i(G))$ from the local distribution of its noisy fitness samples over a wide space $[J_{k,i}^{\min},\ J_{k,i}^{\max}]$. The median value of the fitness samples lying a particular interval, say l, is used as its representative (instead of the average value $(l^{\min}+l^{\max})/2$ of the l-th segment with boundary $[l^{\min},\ l^{\max}]$) because the median value of a frequency distribution is less prone to noisy measurements. A schematic diagram of the expected fitness evaluation of the k-th objective is given in Fig. 4. The entire procedure is executed for k= [1, N] and i=[1, NP].

(d) Spread of the fitness samples: The level of contamination of noise on $J_k(\vec{Y}_i(G))$ can be modeled by the spread $s_{k,i}$ of its sample values away from its expected estimate $\overline{J}_{k,i}$ for k= [1, N]. To calculate $s_{k,i}$, first the median values of fitness samples of each of the L intervals are sorted in ascending order of magnitude. The interquartile range (IQR) of the sorted list of medians is then regarded as a unique measure of $s_{k,i}$ as defined by

$$s_{k,i} = Q_{k,0.75}(\vec{Y}_i(G)) - Q_{k,0.25}(\vec{Y}_i(G)) \ . \quad (29)$$

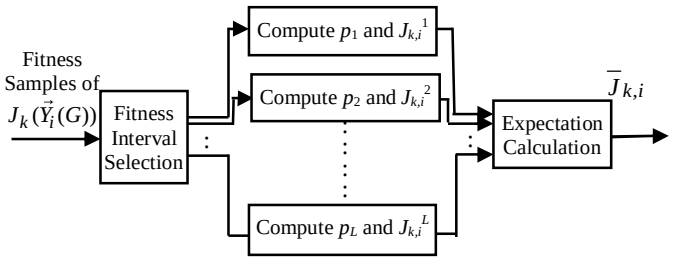

Fig. 4 Expected fitness calculation.

Here, $Q_{k,0.25}(\vec{Y}_i(G))$ and $Q_{k,0.75}(\vec{Y}_i(G))$ respectively symbolize the lower and the upper quartile of the sorted list. IQR is capable to capture the true spread of samples in the noisy environment better than the variance (as in (19)) as the measurement of IQR eliminates the impact of the extreme values of the noisy fitness samples.

The merits of the proposed strategy lie in the following counts. 1) It overcomes the difficulty of the conventional averaging approaches, concerned with referring to the average value of the fitness samples as the effective fitness estimate of a trial solution, presuming equal probability of occurrence of all samples (which may not hold true in the presence of noise). 2) It requires no prior setting of the number of fitness intervals L (as in section IV.B). 3) The non-uniform partitioning of the sample space better captures the noisy local distribution of fitness samples. 4) The expected fitness considering the median values of the fitness samples of each interval provides more robust fitness estimate than representing each interval with its mean value. 5) The spread of the noisy fitness samples is better represented by IQR than the fitness variance.

B. Dominance based Selection by Employed/Onlooker Bees

The strict inequality conditions of deterministic dominance of $\vec{Y}_i(G)$ over $\vec{Y}_j(G)$, as given by the following definitions

(a) $J_k(\vec{Y}_i(G)) \le J_k(\vec{Y}_j(G))$ for k =[1, N] and

(b) $J_l(\vec{Y}_i(G)) < J_l(\vec{Y}_j(G))$ for at least one $l \in [1, N]$

cannot correctly examine the dominance criteria when both $J_k(\vec{Y}_i(G))$ and $J_k(\vec{Y}_j(G))$ are contaminated with noise for k=[1, N]. Deterministic dismissal of apparently inferior candidate solutions from the optimal Pareto front (due to deterministic dominance criteria) may instigate a potential loss of a quality solution in the presence of noise. To circumvent this, a stochastic dominance criterion is adopted for testing the extent of dominance of $\vec{Y}_i(G)$ over $\vec{Y}_j(G)$, denoted by $\vec{Y}_i(G) \prec \vec{Y}_j(G)$, using probability of dominance

$$p(\vec{Y}_i(G) \prec \vec{Y}_j(G)) = \prod_{k=1}^{N} p(\overline{J}_{k,i} \le \overline{J}_{k,j}) \times p(\overline{J}_{k,i} < \overline{J}_{k,j}) \quad (30)$$

where $$p(\overline{J}_{k,i} \le \overline{J}_{k,j}) = 1 - \frac{1}{1+\exp(-c(\overline{J}_{k,i} - \overline{J}_{k,j}))} \quad (31)$$

and $$p(\overline{J}_{k,i} < \overline{J}_{k,j}) = \frac{1}{1+\exp(-c(\overline{J}_{k,j} - \overline{J}_{k,i}))} \ . \quad (32)$$

The proposed probabilistic dominance criterion assists us to maintain the optimal Pareto front up to certain degree of confidence. The Fermi-Dirac probability distributions in (31) and (32) guarantee that

1. For c approaching ∞ and $\overline{J}_{k,j} > \overline{J}_{k,i}$ for k = [1, N], $p(\vec{Y}_i(G) \prec \vec{Y}_j(G)) = 1$ signifying $\vec{Y}_i(G) \prec \vec{Y}_j(G)$.
2. For c approaching ∞ and $\overline{J}_{k,j} < \overline{J}_{k,i}$ for k = [1, N], then $p(\vec{Y}_i(G) \prec \vec{Y}_j(G)) = 0$ signifying $\vec{Y}_j(G) \prec \vec{Y}_i(G)$.
3. If $\overline{J}_{k,j} \cong \overline{J}_{k,i}$ for k= [1, N], then $p(\vec{Y}_i(G) \prec \vec{Y}_j(G)) = 1/4^N$ implying the non-dominance relationship between $\vec{Y}_i(G)$ and $\vec{Y}_j(G)$.

VI. Experiments and Results

The experimental settings used for the comparative study of the relative performance of the proposed algorithm with its competitors along with the performance analysis and the results are summarized in this section. This section includes three experiments, including the analysis of relative performance of ENNSBC over other noisy EMOO algorithms, i) for optimizing noisy benchmark functions and ii) to solve multi-robot box-pushing problem in a) simulation environment and b) real-world platform. The comparative framework for relative performance analysis of the proposed ENNSBC with other existing noisy EMOO algorithms include NNSBC [9], differential evolution for multi-objective optimization with noise−DEMON [6] and non-dominated sorting genetic algorithm-II with α-dominance operator−NSGA-II-A [11]. The parameter settings of the competitor algorithms can be found at http://www.2shared.com/complete/Ez6f6FH9/wcci_2016_ennsbc_supplementary.html.

A. Simulation Results for Noisy Benchmark Function Optimization

Experiments are undertaken to analyze the comparative performance of ENNSBC with its contenders (including NNSBC, DEMON and NSGA-II-A) to optimize noisy versions of 23 CEC'2009 recommended benchmark functions [15] with respect to *hyper-volume ratio* metric. The noisy versions of CEC'2009 benchmark functions are obtained by contaminating the true objective function values with noise samples taken from Gaussian [16], Poisson [17], and Rayleigh [18] distributions. The comparative analysis of the performance of the competitor algorithms can be found at http://www.2shared.com/complete/Ez6f6FH9/wcci_2016_ennsbc_supplementary.html. The reported results reveal that the proposed ENNSBC outperforms other algorithms in a statistically significant fashion.

B. Performance Analysis of ENNSBC for Multi-Robot Box-Pushing Problem

The experiments are carried out in two phases, first by computer simulation on a Pentium machine, and afterward on platform of Khepera-II mobile robots.

(a) Experiments in Simulated Environment

The multi-robot box-pushing problem is implemented in C on a Pentium processor. Experiments are designed to study the

performance of the proposed ENNSBC algorithm over its contenders to handle the noisy MOO in the multi-robot box-pushing problem. In all the experiments, the distance d is contaminated with additive noise samples η such that

$$d \leftarrow d + \eta \tag{33}$$

where η follows certain specific distribution, including Gaussian, Poisson ad Rayleigh.

The structure of a solution vector used in the ENNSBC algorithm is shown in Fig. 5. It begins with an initialization of the current position ($x_{c,t}$, $y_{c,t}$) of the center of mass of the box at the t-th instant and calls the proposed ENNSBC algorithm to evaluate Γ, F_w, F_l, α, r_l and r_w. Then ($x_{c,t}$, $y_{c,t}$) is updated and the incremental energy and time are computed. The process is continued until ($x_{c,t}$, $y_{c,t}$) is close enough to the goal position (x_{gl}, y_{gl}). The pseudo-code for solving the multi-robot box-pushing problem is given in http://www.2shared.com/complete/Ez6f6FH9/wcci_2016_ennsbc_supplementary.html.

Γ	F_w	F_l	α	r_w	r_l

Fig. 5 Structure of a food source (trial solution) in ENNSBC algorithm

The optimized solution of the box-pushing problem for each local movement of the box can be obtained by decoding the best food source from the approximate Pareto front A ($FS(1)$ of ENNSBC) optimizing the energy- and time-objectives of (11) and (15) respectively. It is, however, notable that all food sources in A are equally good (non-dominated). To select the best one among many possible candidates, the following composite measure is considered for each food source $\vec{Y_i} \in A$.

$$J_i = \max(J^*_{1,i}, J^*_{2,i}) \quad \text{for } i = [1, |A|] \tag{34}$$

where $|A|$ is the number of non-dominated solutions in A and

$$J^*_{k,i} = \bar{J}_{k,i} \Big/ \sum_{l=1}^{|A|} \bar{J}_{k,l} \tag{35}$$

represents the normalized estimate of $J_{k,i}^* \in (0, 1)$ for k=[1, 2]. The effective non-dominated food source $\vec{Y_i} \in A$ having the lowest J_i for i= [1, $|A|$] is now identified for decoding the optimal solution (food source) for the single step local movement of the box as obtained by ENNSBC.

In MOO-based simulation of the multi-robot box-pushing problem, the constants K_e in (12) and K_t in (16) are set after some experimentation. K_e and K_t are varied in the range [1, 50] with an incremental step size of 5. It is observed that there is no significant change in performance for $K_e \geq 10$, and $K_t \geq 10$. We have thus fixed K_e = 10 and K_t = 10. Fig. 6 demonstrates an initial configuration of the world map for each of the three arenas, and the starting and the goal positions of the box. We compare the relative performance of our proposed ENNSBC algorithm with NNSBC, DEMON, and NSGA-II-A by varying the noise variance σ^2 in [0.01, 1]. The experiments are repeated for the same three arenas and all the programs are run for 100 times on each arena.

Results of the experiments performed are summarized in Table-I for the first arena (for space economy). Three performance metrics, namely 1) the (average) total energy E consumed by the robots, 2) the (average) total time T required by them, and 3) the (average) total number of steps S (rounded-off to integer) taken by the robots to reach the goal have been used here to determine the relative merits of ENNSBC over other algorithms. The standard deviation of each performance metric obtained by the algorithms is presented within the parenthesis below the respective average value (over 100 runs). It is clear from Table-I that with increasing noise variance, the values of all the three metrics increase significantly. Moreover, it is also evident from Table-I that the jeopardizing effect of noise in deteriorating the performance of the algorithms depends on its distribution. The detrimental effect of Rayleigh noise is the most prominent one, while all the algorithms have performed satisfactorily in the presence of Gaussian noise. However, the results given in Table-I indicate that the performance of the proposed ENNSBC remains consistently better than its competitor algorithms with respect to the energy, the time, and the number of steps required to complete the task for a particular value of σ^2, irrespective of noise distribution. The simulations results for these experiments in three arenas in the presence of noise following Gaussian, Poisson, and Rayleigh distribution (with specific variance) are respectively given in Fig. 7−9. The final configurations of the world maps for three different arenas in presence of measurement noise (of different settings) in Fig. 7−9 clearly reveal that the proposed ENNSBC outperforms the other competitors with respect to the total number of steps taken by the robots to complete the task.

(b) Experiments in Real Environment on Khepera- II Platform

The relative performance of the contender NMOEAs has also been studied on a real-world box-pushing problem using two Khepera-II mobile robots in five different configurations of a world map of 8×6 grids of equal size. The range data of each robot is measured by eight infrared sensors and is represented in a range [0, 1023] corresponding to an obstacle at a distance [2 cm, 5 cm] from the sensor. Each robot is also equipped with one caster wheel and two motor driven side wheels. To realize the box-pushing problem in the real world using noisy MOO formulation, the robots are connected to a Pentium IV personal computer for controlling their motor movements using an NMOEA.

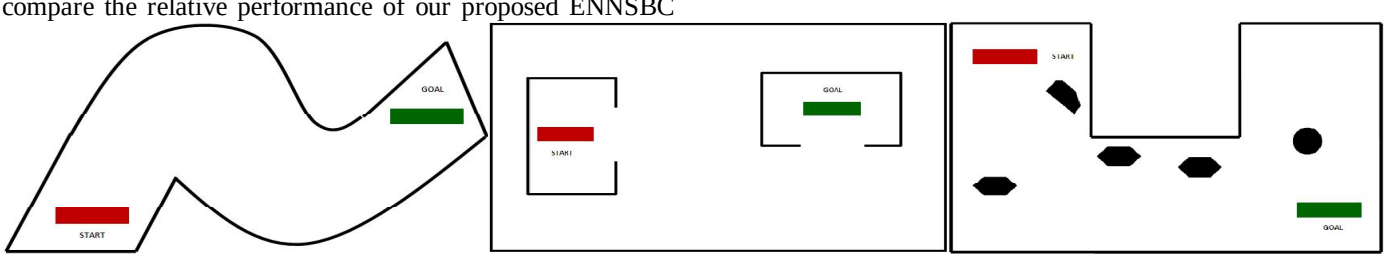

Fig. 6 Initial configuration of the world maps

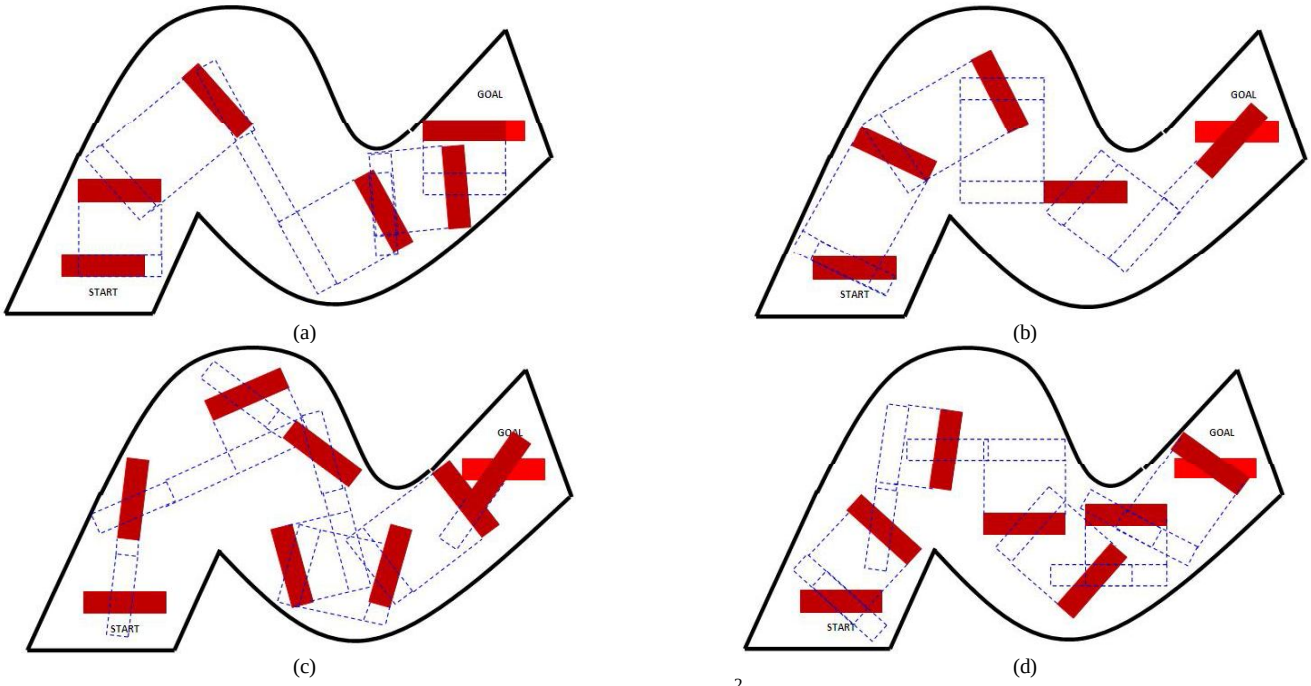

Fig. 7 Path followed by the box in the first arena for zero mean Gaussian noise variance σ^2=0.7 using (a) ENNSBC (b) NNSBC (c) DEMON and (d) NSGA-II-A

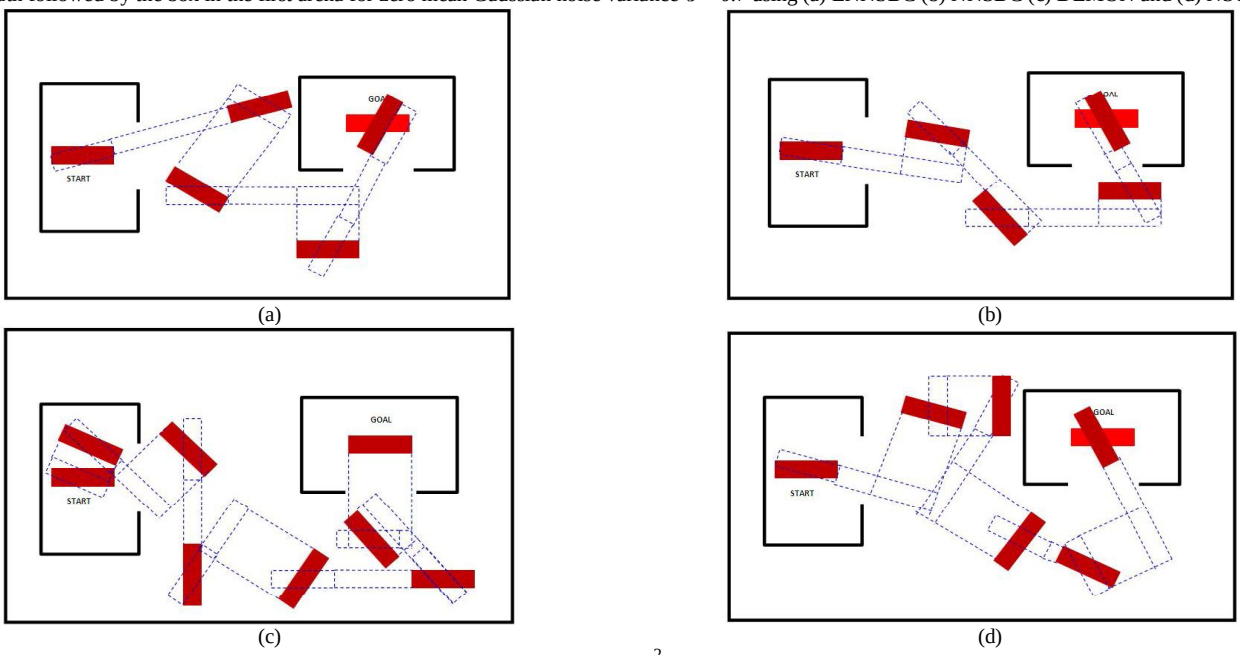

Fig. 8 Path followed by the box in the second arena for Poisson noise variance σ^2=0.45 using (a) ENNSBC (b) NNSBC (c) DEMON and (d) NSGA-II-A

TABLE I-A: PERFORMANCE ANALYSIS OF COMPETITOR ALGORITHMS FOR FIRST ARENA IN PRESENCE OF GAUSSIAN NOISE (BEST METRIC VALUES MARKED IN BOLD)

Noise	Variance	ENNSBC			NNSBC			DEMON			NSGA-II-A		
		E (kJ)	T (sec.)	S	E (kJ)	T (sec.)	S	E (kJ)	T (sec.)	S	E (kJ)	T (sec.)	S
Gaussian	0.2	33.869 (11.816)	**371.79 (25.023)**	**4 (0.011)**	**26.732 (10.450)**	382.07 (26.359)	5 (0.078)	36.151 (12.155)	386.35 (28.054)	5 (0.138)	42.701 (13.229)	421.34 (28.834)	6 (0.162)
	0.4	**26.770 (14.814)**	35.502 (15.305)	**6 (0.229)**	**433.92 (29.724)**	443.62 (32.004)	6 (0.254)	44.491 (16.502)	442.28 (31.449)	6 (0.263)	42.283 (15.808)	434.18 (30.381)	6 (0.337)
	0.6	**30.065 (18.154)**	**463.02 (33.514)**	8 (0.469)	38.298 (18.505)	467.58 (34.213)	**7 (0.380)**	43.928 (19.418)	479.52 (34.469)	8 (0.528)	57.823 (20.617)	579.78 (37.043)	8 (0.547)
	0.8	**33.552 (21.689)**	**598.71 (37.715)**	**9 (0.568)**	43.989 (21.970)	599.51 (38.238)	9 (0.616)	49.475 (22.754)	616.05 (38.436)	9 (0.699)	67.578 (23.919)	630.09 (38.580)	9 (0.753)
	1.0	**41.281 (24.605)**	**609.57 (39.185)**	**10 (0.794)**	44.293 (26.399)	611.00 (40.310)	12 (0.913)	58.876 (27.593)	624.36 (40.610)	11 (0.831)	78.599 (29.153)	657.22 (43.319)	12 (0.934)

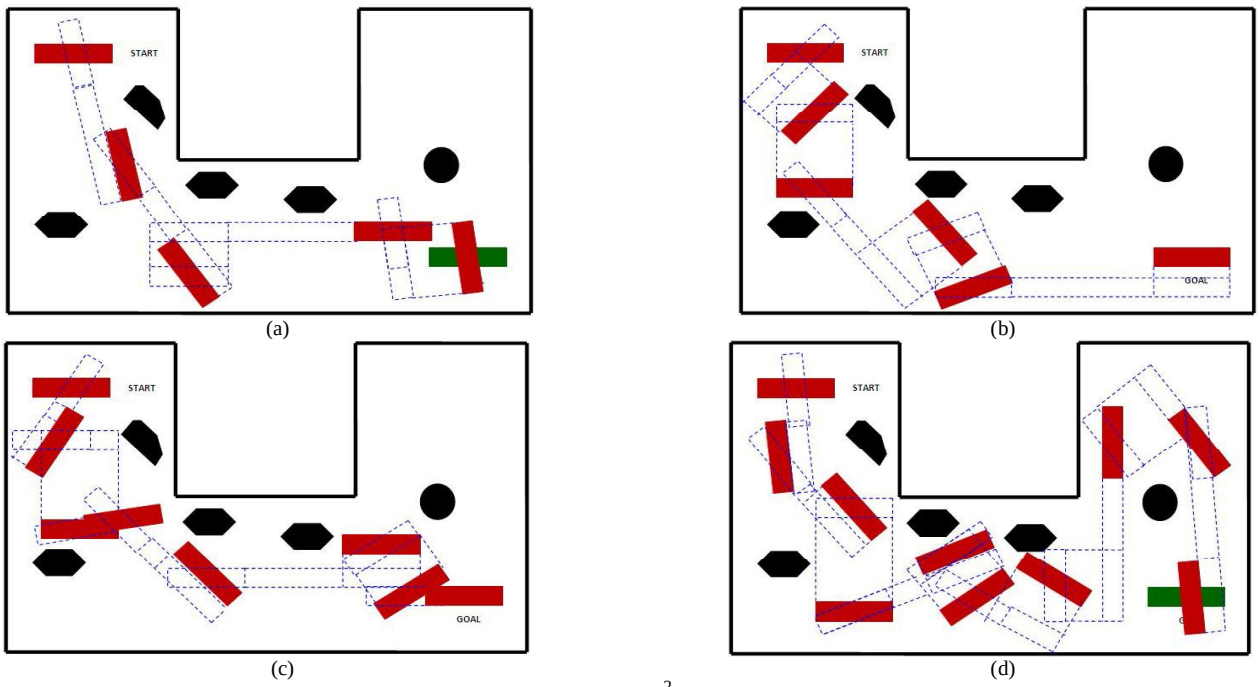

Fig. 9 Path followed by the box in the third arena for Rayleigh noise variance σ^2=0.5 using (a) ENNSBC (b) NNSBC (c) DEMON and (d) NSGA-II-A

TABLE I-B: PERFORMANCE ANALYSIS OF COMPETITOR ALGORITHMS FOR FIRST ARENA IN PRESENCE OF POISSON AND RAYLEIGH NOISE (BEST METRIC VALUES MARKED IN BOLD)

Noise	Variance	ENNSBC			NNSBC			DEMON			NSGA-II-A		
		E (kJ)	T (sec.)	S	E (kJ)	T (sec.)	S	E (kJ)	T (sec.)	S	E (kJ)	T (sec.)	S
Poisson	0.2	**34.667 (10.672)**	**383.57 (25.134)**	**5 (0.054)**	38.023 (11.884)	389.00 (26.822)	6 (0.083)	39.540 (12.412)	396.73 (28.517)	6 (0.149)	49.702 (13.579)	436.67 (29.178)	6 (0.165)
	0.4	**36.835 (15.036)**	442.35 (30.750)	**7 (0.243)**	43.954 (15.329)	**440.90 (29.857)**	8 (0.255)	50.731 (16.254)	469.74 (31.822)	9 (0.285)	55.869 (16.877)	476.84 (32.735)	9 (0.350)
	0.6	**39.711 (18.457)**	**474.56 (34.002)**	**9 (0.442)**	54.401 (18.801)	532.73 (34.249)	9 (0.473)	56.685 (20.361)	563.61 (35.931)	10 (0.530)	65.467 (20.627)	631.96 (37.146)	11 (0.549)
	0.8	**47.368 (21.790)**	**608.60 (37.889)**	**11 (0.585)**	59.595 (22.770)	636.56 (38.506)	11 (0.654)	56.362 (22.219)	626.98 (38.240)	11 (0.748)	83.533 (23.997)	645.11 (38.667)	12 (0.757)
	1.0	**51.194 (25.576)**	**623.25 (39.435)**	**12 (0.814)**	70.075 (26.487)	655.05 (40.403)	12 (0.840)	75.104 (28.126)	658.90 (40.694)	13 (0.917)	84.995 (29.373)	678.31 (43.348)	14 (0.957)
Rayleigh	0.2	**38.936 (11.376)**	400.98 (27.387)	**6 (0.075)**	47.931 (12.112)	**392.48 (25.715)**	6 (0.129)	54.741 (13.073)	424.72 (28.773)	7 (0.152)	66.813 (14.523)	456.01 (29.481)	8 (0.196)
	0.4	**45.736 (15.214)**	**444.64 (30.095)**	**8 (0.251)**	53.497 (15.620)	476.56 (30.781)	8 (0.257)	68.239 (17.692)	482.14 (31.889)	8 (0.311)	64.333 (16.392)	482.69 (33.487)	9 (0.351)
	0.6	**55.894 (18.469)**	**539.80 (34.174)**	**10 (0.450)**	61.466 (19.148)	563.92 (34.427)	11 (0.506)	64.433 (20.542)	579.70 (36.524)	11 (0.538)	68.888 (21.659)	650.69 (37.147)	11 (0.567)
	0.8	65.433 (22.341)	**635.72 (37.952)**	**11 (0.602)**	**58.590 (21.887)**	643.18 (38.356)	12 (0.689)	70.912 (23.088)	645.17 (38.522)	11 (0.751)	86.519 (24.367)	650.71 (38.908)	12 (0.779)
	1.0	**73.837 (26.355)**	**655.63 (39.768)**	**13 (0.825)**	79.482 (27.507)	674.76 (40.405)	14 (0.890)	88.013 (28.872)	678.39 (41.887)	15 (0.929)	90.145 (29.653)	699.68 (43.903)	15 (0.966)

The sensory data of the robots, representing their real-world distances from the obstacles/sidewall of the workspace, are transferred to the connected computer through RocketPort USB Serial Hub-II. Finally, *time division multiple access* is used to transfer the necessary commands to the robots for the effective movement of their motors towards the next position of the box, as determined by the NMOEA running on the connected computer. One sample run of the box-pushing problem realized in the real environment using ENNSBC is given in Fig. 10 in the presence of zero mean Gaussian noise of variance 0.2. Fig. 11 (a) and (b) pictorially represent the evolution of the total energy consumed and the total time taken by the robots, averaged over 50 runs, to execute the complete task. Fig. 11 confirms that ENNSBC outperforms the remaining three algorithms with respect to minimizing both objectives irrespective of noise variance.

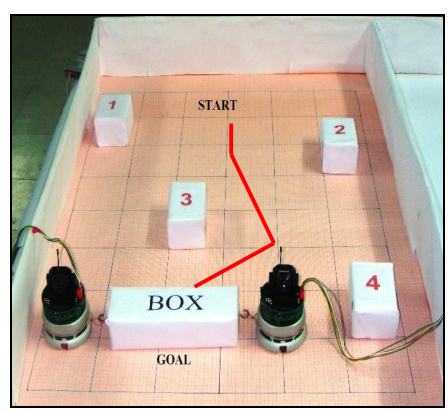

Fig. 10 Path followed by the box by execution of ENNSBC algorithm in Khepera environment with four obstacles and zero mean Gaussian noise variance σ^2=0.2

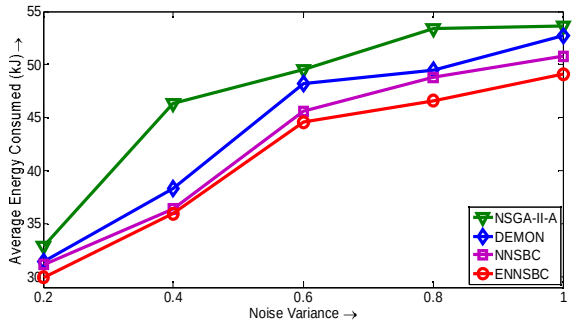

Fig. 11 (a) Plot of average energy consumed by robots (over 50 runs) with zero mean Gaussian noise variance

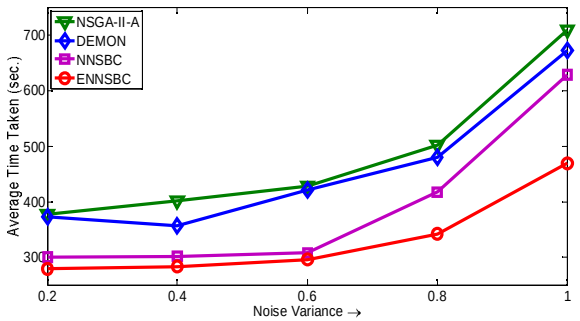

Fig. 11 (b) Plot of average time taken by robots (over 50 runs) with zero mean Gaussian noise variance

VII. Conclusion

The paper studies the scope of noisy MOO in multi-agent robotics using an extension of NNSBC algorithm. The noisy MOO problem is simulated by adding noise (following certain distribution including Gaussian, Poisson, and Rayleigh) with the distance measured by the robot sensors in box-pushing problem. The problem is formulated in multi-objective settings to minimize both the energy consumed, and the time required by the twin robots to transport the box to the goal position.

Experiments have been performed to study the performance of the proposed ENNSBC with its three contenders, including NNSBC, DEMON, and NSGA-II-A. All the experiments are repeated 100 times to determine the average performance of all algorithms with respect to box transfer-time, energy consumption, and the number of steps of planning required to complete the job. The experiments undertaken confirm that with increase in noise variance, the total time spent and the total energy consumed as well as the number of steps required by the robots increase. An intuitive interpretation of this phenomenon is that with increase in noise variance, robots face more constraints to plan local trajectories, thereby increasing the values of the three metrics. For a predefined value of noise variance, the experiments in different workspaces (both in simulation and in real-world environment) reveal that the proposed ENNSBC outperforms its competitors significantly, irrespective of noise distribution.

References

[1] C. R. Kube, and H. Zhang, "The Use of Perceptual Cues in Multi-Robot Box Pushing," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 3, 1996, pp. 2085-2090.

[2] A. Verma, B. Jung, and G. S. Sukatme, "Robot Box-Pushing with Environment-Embedded Sensors," in *Proceedings of 2001 IEEE international Symposium on Computational Intelligence in Robotics and Automation*, 2001, pp. 212-217.

[3] D. Banerjee, P. Rakshit, A. Konar, and R. Janarthanan, "Path-Planning of Mobile Agent using Q-Learning and Real-Time Communication in an Unfavourable Situation", in *Proceedings of IEEE World Congress on Information and Communication Technologies*, October, 2012, pp. 89-94.

[4] P. Bhattacharjee, P. Rakshit, I. Goswami, A. Konar, and A. K. Nagar, "Multi-Robot Path-Planning Using Artificial Bee Colony Optimization Algorithm", in *Proceedings of Nature and Biologically Inspired Computing 2011*: 219-224.

[5] J. Xiao, Z. Michalewicz, L. Zhang, and K. Trojanowski, "Adaptive Evolutionary Planner/Navigator for Mobile Robots", *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, 1997, pp. 18-28.

[6] P. Rakshit, A. Konar, S. Das, L. C. Jain, and A. K. Nagar, "Uncertainty Management in Differential Evolution Induced Multi-objective Optimization in Presence of Measurement Noise", *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 44, no. 7, 2014, pp. 922-937.

[7] P. Rakshit, and A. Konar, "Differential Evolution for Noisy Multi-objective Optimization", *Artificial Intelligence, Elsevier*, Volume 227, 2015, pp. 165-189.

[8] P. Rakshit, and A. Konar, "Non-dominated Sorting Bee Colony Optimization in Presence of Noise", *Soft Computing, Springer*, 2015, DOI 10.1007/s00500-014-1579-z.

[9] P. Rakshit, A. Konar, and A. K. Nagar, "Artificial Bee Colony Induced Multi-objective Optimization in Presence of Noise", in *Proceedings of IEEE Congress on Evolutionary Computation*, 2014, pp. 3176-3183.

[10] P. Rakshit, A. Konar, and A. K Nagar, "Type-2 Fuzzy Induced Non-dominated Sorting Bee Colony for Noisy Optimization", in *Proceedings of IEEE Congress on Evolutionary Computation*, 2015, pp. 1869-1876.

[11] P. Boonma, and J. Suzuki, "A Confidence-based Dominance Operator in Evolutionary Algorithms for Noisy Multiobjective Optimization Problems," in *Proceedings of International Conference on Tools with Artificial Intelligence*, 2009, pp. 387-394.

[12] K. Deb, A. P. S. Agarwal, and T. Meyarivan, "A Fast and Elitist Multi-objective Genetic Algorithm: NSGA II," *IEEE Transactions on Evolutionary Computation*, vol. 2, 1998, pp. 162-197.

[13] M. Babbar, A. Lakshmikantha, D. E. Goldberg, "A Modified NSGA-II to solve Noisy Multi-objective Problems," in *Proceedings of Conference on Genetic Evolutionary Computation*, 2003.

[14] P. Rakshit, and A. Konar, "Extending Multi-objective Differential Evolution for Optimization in Presence of Noise", *Information Sciences, Elsevier*, vol. 305, no. 1, 2015, pp. 56-76.

[15] Q. Zhang, A. Zhou, S. Zhao, P. N. Suganthan, W. Liu, and S. Tiwari, *Multi-objective Optimization Test Instances for the CEC 2009 Special Session and Competition*, Working Report, CES-887, School of Computer Science and Electrical Engineering, University of Essex, 2008.

[16] G. E. P. Box, and M. E. Muller, "A note on the generation of random deviates," *the Annals of Mathematical Statistics*, vol. 29, 1958, pp. 610-611.

[17] D. E. Knuth, *Seminumerical Algorithms*, the Art of Computer Programming, vol. 2, 1969, Addison Wesley.

[18] W. Hörmann, J. Leydold, and G. Derflinger, "General principles in random variate generation," in *Automatic Nonuniform Random Variate Generation, Springer Berlin Heidelberg*, 2004, pp. 13-41.

[19] J. Chakraborty, A. Konar, A. Nagar, and S. Das, "Rotation and Translation Selective Pareto Optimal Solution to the Box-Pushing Problem by Mobile Robots using NSGA-II," in *Proceedings of IEEE Congress on Evolutionary Computation*, 2009, pp. 2120-2126.

[20] P. Rakshit, A. Kumar Sadhu, A. Halder, A. Konar, and R. Janarthanan, "Multi-Robot Box-Pushing Using Differential Evolution Algorithm for Multiobjective Optimization", in *Proceedings of International Conference on Soft Computing and Problem Solving*, vol. 130, 2011, pp. 355-365.