Automatic Calibration of Camera to World Mapping in RoboCup using Evolutionary Algorithms

Patrick Heinemann, Felix Streichert, Frank Sehnke and Andreas Zell
Centre for Bioinformatics Tübingen (ZBIT)
University of Tübingen, Sand 1, 72076 Tübingen, Germany
Email: heinemann@informatik.uni-tuebingen.de

***Abstract*— A common practical problem in mobile robotics is the task to calibrate the robot's sensors. Although, the general mapping of the sensor data to robot-centered world coordinates is given by the hardware configuration, the parameters of this mapping vary even between robots with the same configuration. In the RoboCup domain, these parameters can change drastically after transport or physical contact during game play. It is therefore necessary to recalibrate the robots for their next assignment within a few minutes, not only in order to fulfill the future regulatory requirements of the RoboCup organization committee to keep the setup time as low as possible. As camera systems, especially omni-directional systems, are currently the most important sensors in RoboCup, a reliable and fast calibration method for the mapping of image to world coordinates is necessary. Since the RoboCup environment, i.e. the soccer field, has known dimensions and is also static, automatic calibration using the features and landmarks of the soccer field is possible if the robot is given an image from a known pose. In this paper, an efficient evolutionary approach to automatic camera calibration is presented, which is independent of the hardware configuration. It only requires a quality function for the parameter settings, which allows lazy evaluation. To meet the time constraints given for this real-world optimization problem, a novel mutation operator is introduced to enhance the performance of the evolutionary algorithm. It samples a number of alternative solutions using a high rate of lazy evaluation, before deciding on the true mutative change applied on the given individual. This new mutation operator proves to be fast and most reliable on the camera to world calibration problem.**

I. Introduction

One of the major tasks for a mobile robot is the detection of objects and landmarks in its environment. Self-localization algorithms and the planning of collision-free paths are depending on a model of the robot's environment resulting from these features. A consistent model of the complete surroundings of the robot can only be extracted using a sensor that delivers three-dimensional data. Unfortunately, such sensors are rare and expensive and most of the current mobile robots use vision systems as their main sensor. Without using a complex stereo camera system, there is no way of extracting a complete three-dimensional environment model from a two-dimensional camera image.

In many applications of mobile robots, especially in the RoboCup domain [9], it is sufficient to build a two-dimensional model of the floor plane where the robot moves, including all obstacles and landmarks. Thus, an environment model can be generated by mapping the image coordinates onto two-dimensional world coordinates on the ground floor. With such a mapping the robot can extract objects and landmarks in the image and map them to world coordinates using their contact point to the floor [4]. The resulting model is similar to a bird's eye view from above the field.

For typical camera systems this mapping can be roughly approximated by collecting corresponding coordinate pairs that describe the same feature, followed by fitting a two-dimensional function through these correspondences. However, this approximation procedure is very time consuming and prone to errors. A better way to find the mapping is to use camera calibration techniques that extract the extrinsic and intrinsic parameters of the camera system and build a complete physical model of the projection [14]. Although this method results in a very accurate model of the mapping, it is time consuming, too. Furthermore, this technique is only used for perspective cameras.

Omni-directional camera systems are composed of a perspective camera pointing upwards onto a convex shaped mirror that reflects the complete surrounding of the robot (cf. Figure 1). Thus, the camera of an omni-directional camera system can also be calibrated using the algorithm of Tsai [14]. Knowing the calibrated parameters of the camera and the exact position of the camera and the mirror on a robot, a special mirror shape can then be calculated to map the ground floor without distortion [6], [10], [11].

Even if the shape of the mirror is known or especially designed, the displacement of the mirror in all three translational axes influences the mapping from image to world

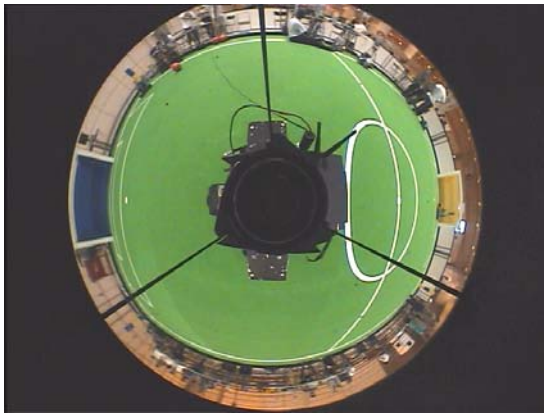

Fig. 1. Distorted image of an overhead omni-directional camera.

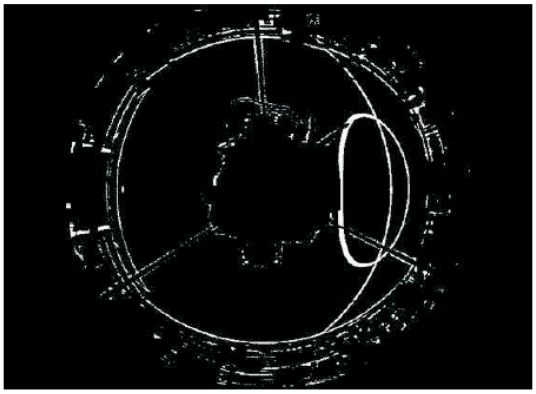

Fig. 2. All extracted white points from the camera image given in Fig. 1.

coordinates, too. However, after transport or maintenance of the robot these parameters change, often resulting in a wrong mapping. Then, a time-consuming recalibration or readjustment of the camera system is required using these approaches. Therefore, a calibration method is needed, that is able to automatically estimate the parameters of the vision system in short time. Especially RoboCup robots must recalibrate their cameras in-between their games, as a consequence of physical contact to other robots during a match.

In this paper we present an approach to automatic calibration of the camera to world mapping using an image taken by the omni-directional camera system of a RoboCup robot located at a predefined position on the field. In contrast to the semi-automatic method presented in [1] which uses a special reference pattern as feature input, our approach is based on the landmarks and features that are already contained in the robot's environment.

We address this automatic camera calibration problem by means of evolutionary algorithms, because they incorporate significant advantages for the given optimization problem. On the one hand, evolutionary algorithms are global optimization approaches, able to deal with multi-modal fitness functions. On the other hand, they allow problem specific extensions to fit the optimization algorithm to the problem at hand. Both features are utilized in this paper, resulting in a feasible algorithm for the given time constrained real-world optimization problem of camera to world calibration.

The following section gives an introduction to the camera to world coordinates calibration problem. Then, the applied evolutionary algorithms are outlined in Sec. III, which also introduces the novel mutation operator based on lazy evaluation using multiple mutant alternatives. Sec. IV introduces the hardware and the field settings used for the experiments in Sec. V where the optimization problem is applied to a real RoboCup robot. Finally, Sec. VI gives our conclusions on the results obtained.

II. Camera to World Coordinates Mapping

Depending on the hardware configuration, the camera to world coordinate mapping functions can be very different. In the context of RoboCup, cameras pointing upwards into a convex mirror have become a common hardware configuration. The resulting image of such an omni-directional camera system is a distorted bird's eye view of the environment, see Fig. 1. One advantage of omni-directional camera systems in the context of image to world mapping is the radial symmetry of the system. Instead of a two-dimensional mapping

$$d:\ \mathbb{N}^2 \mapsto \mathbb{R}^2,\ (r_i, \phi_i) \rightarrow (r_w, \phi_w) \tag{1}$$

of polar image coordinates (r_i, ϕ_i) originating in the center of the image to robot centered world coordinates (r_w, ϕ_w), this symmetry simplifies the mapping to a one-dimensional mapping of the radius

$$d:\ \mathbb{N} \mapsto \mathbb{R},\ r_i \rightarrow r_w, \tag{2}$$

as the angle ϕ is not changed from image to world coordinates in such a system. Preliminary experiments showed, that the quality of the mapping does not change significantly when using a one-dimensional mapping function instead of a two-dimensional. Even if the optical axis of the camera and the symmetry axis of the mirror are not collinear, a one-dimensional mapping function gives good results, if this displacement is a parameter of the function d.

A suitable mapping between image coordinates and robot-centered world coordinates is evaluated by comparing the resulting mapping of a number of reference points to their true coordinates. This can be done by placing the robot in the center of a checker board grid of known dimensions for calibration. Due to the reduced time for preparation in a tournament, this is no longer possible. But the known dimensions of the soccer field can be used as reference grid instead of an artificial grid, if the robot is given his correct position for calibration. This approach allows an on-site camera calibration for RoboCup robots.

Unfortunately, such a scenario limits the application of classic approaches for two reasons: First, in this on-site calibration, the reference points (i.e. the white points, see Fig. 2) are not evenly distributed over a wide range of distances, thus leaving too few intermediate points for linear interpolation methods. Secondly, due to the limited amount of reference points, noise has a significant effect on the results of more sophisticated methods like B-splines, which are likely to be subject to over-fitting. Therefore, we decided to map the image coordinates onto world coordinates using following parameterized mapping function:

$$d(x, r) = (x_0 \cdot e^{x_1 + x_2 r^2} + x_3 \cdot r^2 + x_4 \cdot r + x_5) \cdot x_6, \tag{3}$$

with r being a function that is used to correct the offset of the symmetry axis of the mirror to the optical axis as follows:

$$r(x, (r_i, \phi_i)) = \sqrt{(x_7 - r_i cos\phi_i)^2 + (x_8 - r_i sin\phi_i)^2}. \tag{4}$$

The mapping function given in Equ. 3 is mainly influenced by a polynomial term. However, as the mapping of the omni-directional camera system includes the horizon, we decided to approximate the infinite slope of the real mapping function through an exponential term. This results in a decision vector x of nine parameters, which are to be optimized.

To validate that a pixel $p = (r_i, \phi_i)$ observed in the camera image corresponds to a white field line, it is transformed into world coordinates $(d(x, r), \phi_i)$ and it is checked whether or not $(d(x, r), \phi_i)$ is white in the model of the soccer field:

$$C(p) = \begin{cases} 1 & : \text{ if point } M(d(x, r), \phi_i) \text{ is white} \\ 0 & : \text{ else.} \end{cases} \quad (5)$$

For all white pixels $\{p_n = (r_{i,n}, \phi_{i,n}) \mid n \in [1 \dots N]\}$, which are below an artificial horizon given by $d(x, r_n) < d_{max}$ this check has to be performed, resulting in the fitness function:

$$f(x) = \frac{\sum_{n=0}^{N} r_{i,n}}{\sum_{n=0}^{N} (r_{i,n} \cdot C(p_n))} - \delta. \quad (6)$$

Weighting C with $r_{i,n}$ is necessary, because close white points are typically overrepresented in a reference image. This fitness function is to be minimized and converges to negative fitness values. The additional constant δ has been introduced to give suitable selection pressure for the particle filter approach and remained in the fitness function ever since. However, it has no negative effect on the other optimization algorithms. For the experiments presented in this paper a value of $\delta = 8$ was chosen.

Although there are up to 50,000 white points in a given reference image, not all these points need to be tested. This is exploited using lazy evaluation [5], which reduces the execution time, but also introduces noise into the fitness function. A lazy evaluation of $L = 10\%$ results in randomly chosing only $5,000$ white points for testing. This subsampling extremely reduces the computational load. Throughout this paper, the amount of lazy evaluation is given as L where it is applied. Please note, that although the optimization is recorded over the number of fitness evaluations, an individual evaluated using lazy evaluation requires only a fraction of the time for a full fitness evaluation.

III. The Optimization Algorithms

To solve the given real-world optimization problem we applied three different optimization algorithms: a generational genetic algorithm (GA) [7], an evolution strategy (ES) [12] and a particle filter approach (PF) [2]. The GA was a generational real-valued GA using a real-valued vector as genotype and applying non-isotropic self-adaption mutation [12] and a discrete real-valued one-point crossover with $p_m = 1.0$ and $p_c = 0.5$. The GA used an elite group size of one and tournament selection for parent selection with a tournament group size of 4.

The ES applied the same mutation and crossover operators and parameters, but was based on a (μ, λ)-strategy using greedy best selection for environmental selection, i.e. down selecting from λ to μ, and random parent selection. Due to the comma-selection the ES had no elite population.

The PF approach we applied, used mutation only and a particle wheel selection method to select the parents. This particle wheel selection method resembles the probabilistic roulette-wheel selection, but uses as many equidistant pointers as individuals that are to be selected. This way random fluctuations in the composition of the selected individuals are reduced, but its selection pressure depends on the range of the fitness values, thus the additional constant in Equ. 6.

The non-isotropic self-adaption mutation operator uses one parameter σ to control the mutation steps size:

$$\sigma' = \quad \sigma \cdot e^{\tau \cdot N(0,1)} \quad (7a)$$

$$x_i' = \quad x_i + \sigma' \cdot N_i(0, 1). \quad (7b)$$

For the following experiments we used a value of $\tau = 0.15$ and due to the implementation given in JavaEvA [13] this mutation operator acted on a normalized search space. Because σ is subject to mutation and indirect selection, it is able to self-adapt to the local properties of the search space. Still, the initial value σ_{init} is an important parameter, since a small value for σ_{init} may enhance fast convergence, while a larger value for σ_{init} enables the optimizer to escape local optima in the initial phase of the optimization process.

A. The Lazy Particle Mutation Operator

A novel mutation operator is introduced in this paper, which combines features of the self-adaptive mutation operators of real-valued ES and the search strategy of particle filters. This *lazy particle mutation operator* works as follows: Instead of generating just one solution alternative x', it generates p solution alternatives x'^i (particles) according to Equ. 7 and evaluates these alternatives taking the full advantage of lazy evaluation. Then the lazy particle mutation operator choses the best mutant x'^* as true mutation, which is then allowed to enter the traditional evolutionary cycle.

The lazy particle mutation operator is in the following referred to as (l, p)-mutation, l giving the amount of local lazy evaluation for the particles and p giving the number of particles used. This lazy particle mutation operator could also be interpreted as using a local model of the search space, which relates the lazy particle mutation operator to model-assisted evolutionary algorithms [8]. The local model built for the lazy particle mutation operator could be considered a simple nearest neighbor model based on the lazily evaluated particles, which is thus quite unreliable due to noise. Building a more sophisticated model of the search space is an interesting alternative, which might counter the effect of noise. However, the time constraints of this real-world optimization problem prevented any extension in that direction.

IV. Hardware and field settings used for the experiments

Three types of robots were used for the following experiments, two generations of robots of our Attempto Tübingen Robot Soccer Team and one robot of the Mostly Harmless RoboCup team. All robots are equipped with an omnidirectional camera system consisting of a perspective camera pointing upwards to a convex mirror. Robot I is from our previous RoboCup team and has a standard 25fps camera with a resolution of 768×576 pixels and 5bit resolution per

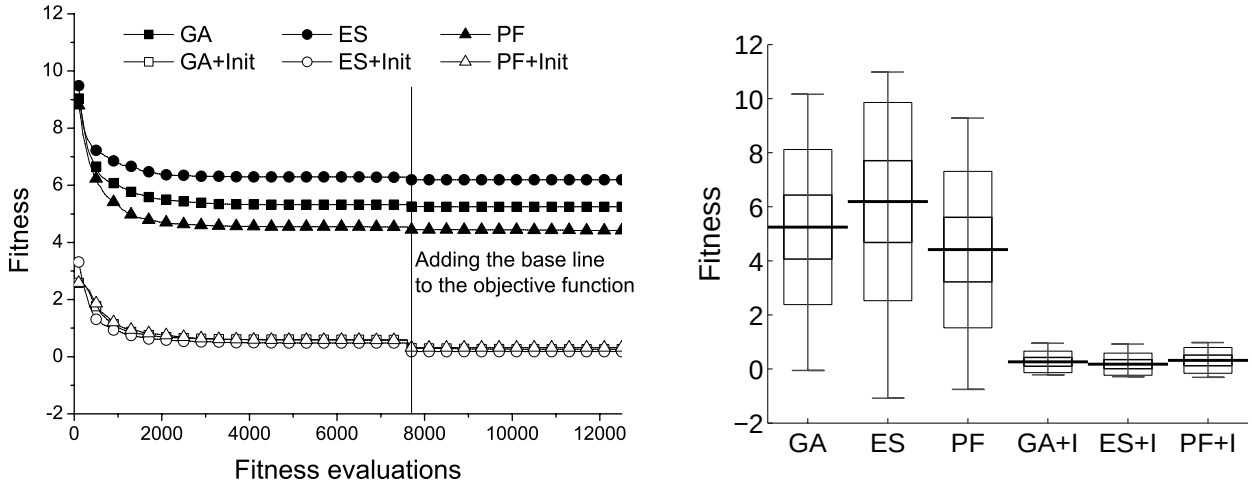

Fig. 3. GA, ES and PF on the camera to world calibration problem using either random initialization or initialization based on previous parameters.

color channel. Robot II of our new team is equipped with a 50fps, 580×580 pixel camera with 8bit resolution per color channel. Robot III is a robot of the Mostly Harmless team and is equipped with a camera comparable to that of robot II but has a different mirror. In addition to the three different robots and camera systems, we also tested the algorithm on different RoboCup fields. The first one (field I) is a training field of the Mostly Harmless team of size $6 \times 5m^2$, which is not large enough to fulfill the requirements of current RoboCup fields. For the second and third setting, images from the RoboCup World Cup 2004 at Lisbon (field II) and from the RoboCup German Open 2005 at Paderborn (field III) with a field size of $12 \times 8m^2$ were used.

V. Experimental Results

The performance of the algorithm is demonstrated using an exemplary image taken by robot I on field II, which was to be mapped onto the standard RoboCup tournament field. The final solution, however, is shown to be applicable to several hardware and field configurations.

Preliminary experiments indicated that the two parallel lines of the penalty area and the goal line introduced a very deceptive local optimum. Especially, if the initial slope of the mapping function in Equ. 3 is too high, the goal line is often mapped onto the penalty base line, introducing a deceptive local optimum. Therefore, the penalty base line was initially not included in the fitness function, but was introduced after about $2/3$ of the fitness evaluations, which causes a small step in the fitness plots.

For the given optimization problem the range of the decision variables was: $x_0 = [20; 25]$, $x_1 = [-10; -2]$, $x_2 = [60; 100]$, $x_3 = [1; 25]$, $x_4 = [30; 70]$, $x_5 = [-1; 1]$, $x_6 = [0.05; 0.3]$, $x_7 = [-0.02; 0.02]$, and $x_8 = [-0.02; 0.02]$.

To give statistically sound results for each algorithm setting, the results were averaged over 100 multi-runs and 2,500 fitness evaluations, if not mentioned otherwise. Additionally, for some experimental results not only the mean fitness, but also the standard deviation, the extreme values of the fitness values and the 95%-confidence interval of the mean are given.

A. Initial Results

In an initial set of experiments using only 20 multi-runs for 12,500 fitness evaluations we compared the GA and the PF approach with a population size of 50 to a $(2, 50)$-ES using $\sigma_{init} = 0.02$ on the camera to world calibration problem. Fig. 3 compares the performance of the three algorithms without and with problem specific initialization based on the most recent parameters of the mapping function. This initialization is based on the assumption that the change in the parameters due to transport or repairs of the robot are likely to be minor. This is because neither the curvature of the mirror nor will the overall distance between the camera and the mirror will change dramatically for these events.

Indeed the results given in Fig. 3 confirm this assumption. The arbitrary initialization performs worse and is unable to equal the performance of the most recent parameter setting. Using the most recent parameter setting as initial solution on the other hand enables the optimization algorithms to improve the given initial solution significantly and reliably. Experiments also illustrated the practical need for recalibration of the camera to world mapping after transport or repair. However, the best solutions obtained by the randomly initialized populations are significantly better than the ones obtained using the most recent parameters as initial values. Thus, the optimization problem is likely to be multi-modal. And because we decided to keep the problem specific initialization, we needed to make the optimization algorithm less prone to premature convergence.

Therefore, the first parameter we investigated was the population size or the ES parameter λ, respectively. But since the previous experiments proved to be too time consuming for the given application, we introduced $L = 10\%$ for lazy evaluation at this point. A more detailed discussion of the effect of lazy evaluation is given in Sec. V-B. Fig. 4 shows that the population size is an important parameter and too

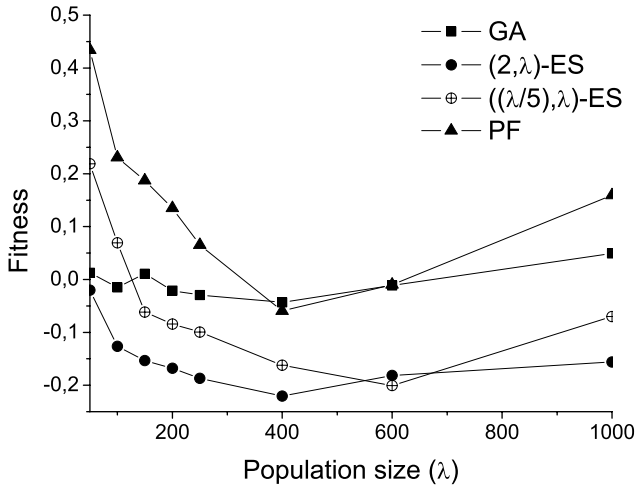

Fig. 4. Population size for GA, ES and PF using $\sigma_{init} = 0.02$, $L = 10\%$.

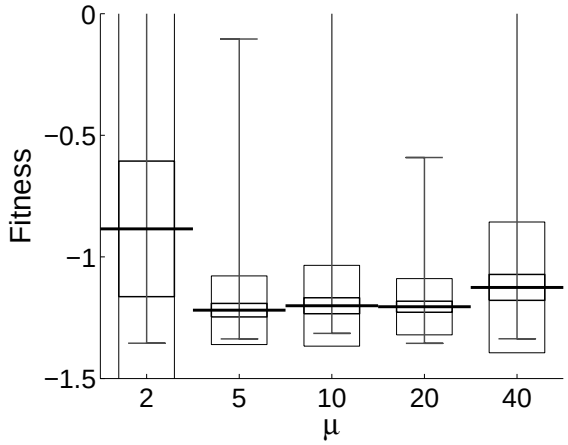

Fig. 5. Effect of μ on the $(\mu, 400)$-ES using $\sigma_{init} = 0.2$, $L = 10\%$.

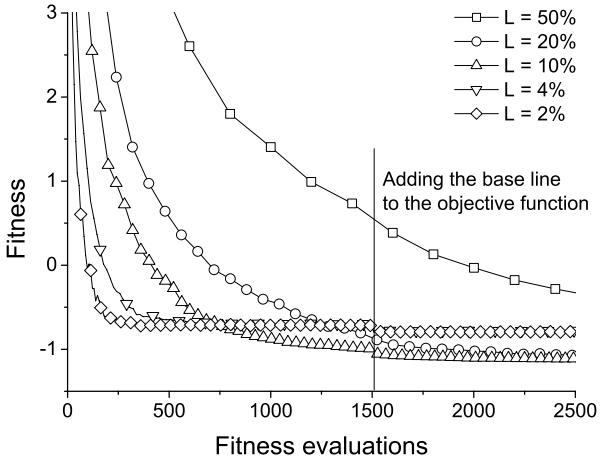

Fig. 6. Impact of lazy evaluation L on the $(5, 400)$-ES using the non-isotropic mutation with $\sigma_{init} = 0.2$.

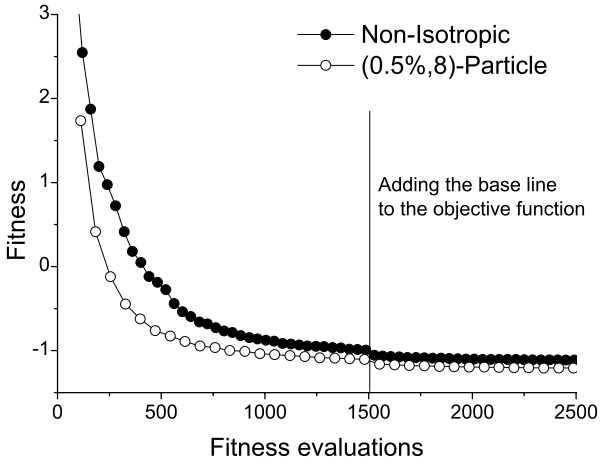

Fig. 7. Comparing the non-isotropic to the non-isotropic lazy particle mutation using $(5, 400)$-ES, $\sigma_{init} = 0.2$, and $L = 10\%$.

small population sizes lead to premature convergence, while too large population sizes reduce the speed of convergence. Based on these experiments, we have chosen the (μ, λ)-ES and $\lambda = 400$ for the following experiments.

The results indicate that the optimization problem at hand is truly multi-modal, thus we evaluated the effect of μ on the $(\mu, 400)$-ES and also raised $\sigma_{init} = 0.2$ to enable the ES to escape local optima in the initial optimization phase. Fig. 5 shows that the ES with increased σ_{init} performs significantly better and that $\mu > 2$ is sufficient to reduce the chance of premature convergence significantly. Therefore, we decided to use a $(5, 400)$-ES and $\sigma_{init} = 0.2$ from now on.

B. The Effect of Lazy Evaluation

Since the optimization problem allows lazy evaluation and the time constraint is important, we decided to evaluate the effect of lazy evaluation on the camera to world calibration problem based on the $(5, 400)$-ES. Fig. 6 shows that lazy evaluation is able to increase the speed of convergence significantly. Using $L < 5\%$ enables the ES to converge to sub-zero fitness values within only 500 fitness evaluation equivalents. Unfortunately, for $L < 5\%$ the optimization algorithms is also subject to limited convergence due to the noise introduced by lazy evaluation.

Therefore, we are faced with the dilemma of having a reliable calibration algorithm, which is too slow for a practical application, or having a fast but unreliable calibration algorithm. We were able to address this dilemma by introducing the novel lazy particle mutation operator.

C. The New Mutation Operator

Although there is a clear positive effect of global lazy evaluation, there is also a severe limitation. Lazy evaluation introduces noise to the fitness function, which can limit the ability of the ES to converge to the true optimum. Thus, instead of increasing the global lazy evaluation L to values smaller than 10%, we applied the new lazy particle mutation operator.

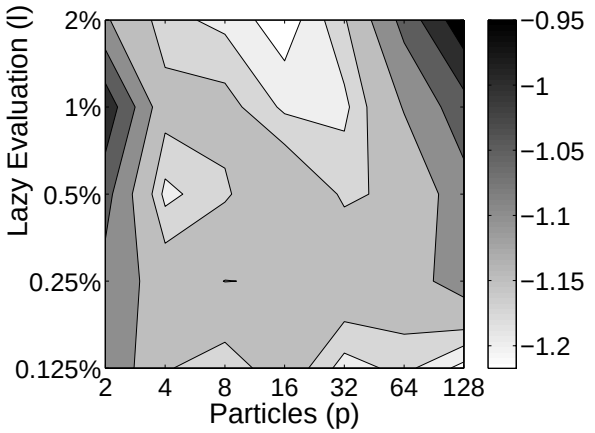

Fig. 8. Grid search on the parameters l and p using a $(2, 100)$-ES, $\sigma_{init} = 0.2$, and $L = 10\%$. Optimal parameters are $p > 32$ and $l = 0.125\%$ or $p = 16$ and $l = 2\%$.

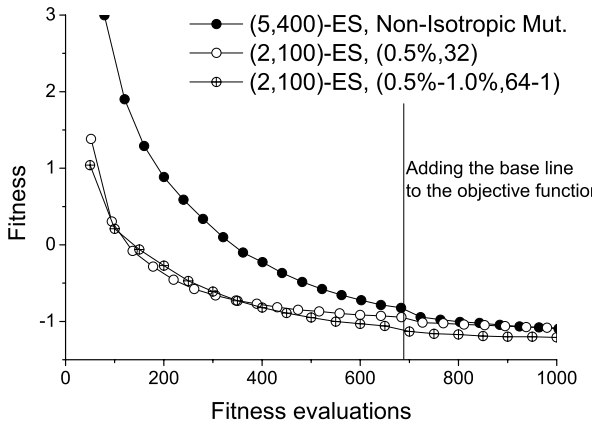

Fig. 9. Comparing the standard ES to an ES with static and dynamic lazy particle mutation using $\sigma_{init} = 0.2$, and $L = 10\%$.

Due to its local sampling technique based on a high amount of lazy evaluation $l \ll 10\%$, it is able to search more efficiently and escape local optima, because of the multiple mutation candidates evaluated. At the same time the resulting mutant is only subject to a limited amount of noise, because the global lazy evaluation L for the true fitness function evaluation is larger than the local lazy evaluation factor l for evaluating the particles.

See Fig. 7 for an initial experimental result using a $(5, 400)$-ES with $\sigma_{init} = 0.2$, and $L = 10\%$ comparing the standard mutation to the $(0.5\%, 8)$-lazy particle mutation operator. The lazy particle mutation operator converges significantly faster in the initial optimization phase, is less prone to premature convergence to local optima and converges to better solutions in the long run.

Even within the first two generations the ES using the lazy particle mutation operator outperforms the standard mutation. Please note, that the fitness plot starts after the first generation has been evaluated and mutated and the next generation has been selected. Thus, the different start values in Fig. 7 for the ES with and without lazy particle evaluation result from the logging mechanism in JavaEvA and not from different initial populations.

The more efficient lazy particle mutation operator and the fact that this mutation operator is less prone to premature convergence allowed us to apply a more greedy $(2, 100)$-ES, which has proven to be infeasible for the standard mutation operator, see Fig. 4 and Fig. 5.

The new parameters l and p introduced by the new lazy particle mutation operator are now subject to a detailed grid search. For this experiment a $(2, 100)$-ES is used with 2,500 fitness evaluations using $\sigma_{init} = 0.2$ and $L = 10\%$, while varying l from 0.125% to 2.0% and p from 2 to 128.

Fig. 8 illustrates that the performance of the lazy particle mutation operator decreases with decreasing number of particles regardless of the amount of lazy evaluation applied and proves thus the general concept of the lazy particle mutation operator. This experiment also shows that too many particles with reduced lazy evaluation cause the optimization algorithms to become too slow to finish successfully within 2,500 fitness evaluations. Additionally, Fig. 8 shows that there are two configurations where the lazy particle mutation operator is successful:

- Higher lazy evaluation $l < 0.5\%$ and many particles.
- Lower lazy evaluation $l \geq 0.5\%$ using fewer particles.

Both parameters seem to be correlated and point to two different effects of the lazy particle mutation operator.

First, a high number of particles and much lazy evaluation allows the lazy evaluation operator to increase the initial convergence rate and to reduce the chances of premature convergence to a local optimum in the initial optimization phase. At the same time it can impair the final convergence phase due to the noise that affects the mutants generated. Secondly, a medium number of particles with less lazy evaluation still improves the initial convergences phase to some extent, but at the same time converges more successfully to the optimum in the final optimization phase.

In this final phase of convergence, i.e. when the optimization has escaped all local optima and has reached the basin of attraction of the global optimum, a localized search with less noise is more suited to identify the correct parameters than a global noisy search strategy. This effect has also been described by Fernàndez *et al.* [3]. In their paper they reduced the population size of a GA gradually to move from a global search to a local search to save computational resources.

Due to the given time constraints, we applied the same strategy to gradually move from global search to local search during the optimization process. But instead of changing the population size, we moved from higher lazy evaluation $l < 0.5\%$ and many particles to lower lazy evaluation and no particles. For this experiment the number of fitness evaluations was further reduced to 1,000 evaluations per

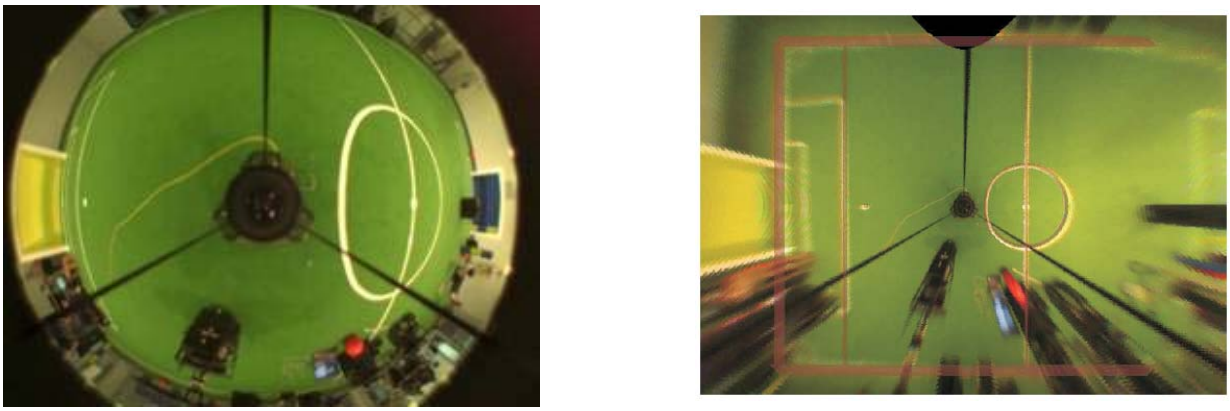

Fig. 10. Example image of robot II on field III (left) and the correctly transformed image (right) after automatic calibration performed within 10-20 seconds using a C++ implementation of the proposed optimization algorithm on an Athlon XP 2400+.

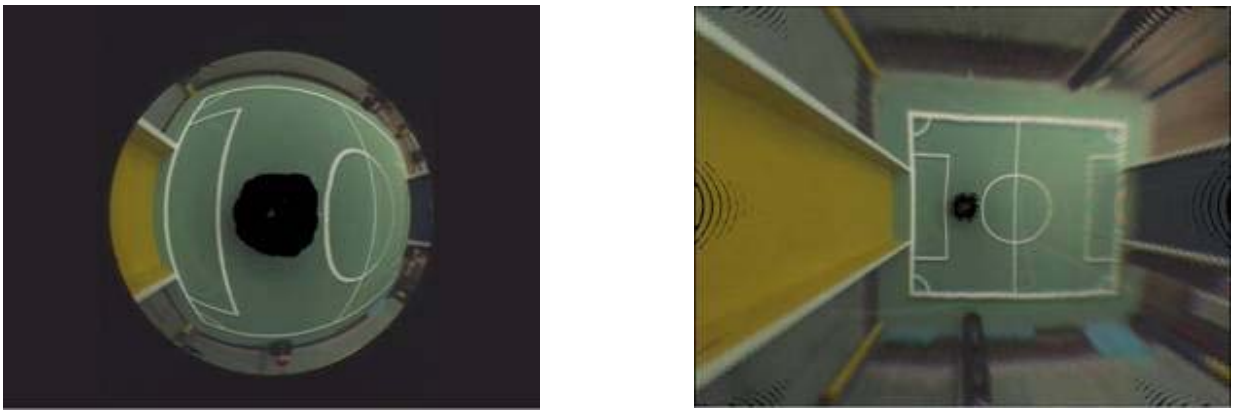

Fig. 11. Example image of robot III on field I (left) and the correctly transformed image (right) after automatic calibration performed within 10-20 seconds using a C++ implementation of the proposed optimization algorithm on an Athlon XP 2400+.

optimization run, to reflect the real-world time constraint for the given application. We compared the standard $(5, 400)$-ES to a $(2, 100)$-ES using either static $(0.5\%, 32)$-lazy particle mutation or dynamic $(0.5\%-1.0\%, 64-1)$-lazy particle mutation. The dynamic lazy particle mutation operator changed the parameters l and p linearly each generation to move from the initial values $l_s = 0.5\%$ and $p_s = 64$ to the final values $l_f = 1.0\%$ and $p_f = 1$. The final values of the dynamic lazy particle mutation actually give the standard mutation operator, because there is only one particle. All algorithms used $\sigma_{init} = 0.2$, and $L = 10\%$.

Fig. 9 shows that both the static and the dynamic lazy particle mutation ES approaches outperform the standard ES in the initial optimization phase. Both approaches using the lazy particle mutation converge faster and are at the same time less prone to premature convergence. However, the standard ES is able to equal the performance of the static lazy particle mutation ES in the final phase of the optimization process, because the static lazy particle mutation operators introduces too much noise to allow an efficient local search in the final optimization phase. This is not the case for the dynamic lazy particle mutation operator. Here the final convergence phase is not impaired due to noise, therefore it is able to converge smoothly to very good solutions. Fig. 10 and Fig. 11 show the results of the algorithm when applied to different types of robot hardware.

VI. Conclusions

Because of the practical need for an on-site automatic camera to world calibration algorithm for RoboCup tournaments, we have evaluated the application of evolutionary algorithms. Unfortunately, the traditional evolutionary approaches proved to be to slow, unreliable and too time consuming for the given real-world parameterization problem. Even utilizing lazy evaluation the performance of the standard approaches was not sufficient. Therefore, we have developed a novel mutation operator called the lazy particle mutation, which allows us to meet the time constraints.

This new mutation operator utilizes lazy evaluation and the idea of multiple sample points from particle filters to find the most suitable mutation candidate as resulting mutant. The (l, p)-lazy particle mutation operator generates p mutation candidates, evaluating each candidate with $l\%$ of lazy evaluation and choosing the best mutation candidate as final mutant. The lazy evaluation of multiple mutation candidates gives a very simple model of the local search space and thus resembles concepts developed in the field of model-assisted EA approaches. However, the simple strategy of the lazy

particle mutation operator can be applied to any type of problem allowing lazy evaluation even if it does not allow model building, e.g. in case of automated programming or combinatorial problem instances. Additionally, the lazy particle mutation strategy proposed in this paper can be applied to virtually any random mutation operator regardless of the data type used.

Applying the lazy particle mutation operator enabled us to solve the given problem more efficiently. On the one hand due to the increased speed of convergence in the initial optimization phase and on the other hand due to the reduced chance of premature convergence. Still, the lazy particle mutation impaired the final optimization phase, due to the noise introduced on the mutants. A grid search on the two parameters l and p confirmed this interpretation. Using the lazy particle mutation with dynamic parameters resolves this problem. It allowed us to reduce the number of required fitness evaluations further and enabled the practical application on RoboCup tournaments. This resulted in a very fast and reliable automatic camera to world calibration algorithm by means of evolutionary algorithms based on the lazy particle mutation operator.

Future research will analyze the lazy particle mutation operator in more detail on artificial benchmark functions, regarding whether or not the results presented here can be generalized to other types of optimization problems, including uni-modal optimization problems. On these problem instances the effect of the alternative selection methods for the particles of the mutation operator including the utilization of local model building should be investigated, which was unfortunately not possible for the given optimization problem. Additionally, the new mutation operator should be compared to model-assisted evolutionary approaches on expensive optimization problems that allow lazy evaluation, since the similarity between the two approaches is high.

REFERENCES

[1] G. Adorni, M. Mordonini, S. Cagnoni, and A. Sgorbissa. Omnidirectional stereo systems for robot navigation. In *2003 Conference on Computer Vision and Pattern Recognition Workshop*, volume 7, pages 79–89, 2003.

[2] A. Doucet, N. de Freitas, and N. Gordon, editors. *Sequential Monte Carlo methods in Practice*. Springer-Verlag, 2001.

[3] F. Fernàndez, E. Cantù-Paz, J. I. Lòpez, and T. Manzano. Saving resources with plagues in genetic algorithms. In X. Yao, editor, *8th International Conference on Parallel Problem Solving from Nature (PPSN VIII)*, pages 272–282. Springer-Verlag, 2004.

[4] P. Heinemann, Thomas Rückstieß, and Andreas Zell. Fast and Accurate Environment Modelling using Omnidirectional Vision. In U. Ilg, H. Bülthoff, and H. Mallot, editors, *Dynamic Perception 2004*. Infix Verlag, 2004.

[5] P. Henderson and J. M. Morris. A lazy evaluator. In *3rd Annual ACM symposium on Principles of Programming Languages*, pages 95–103. ACM, 1976.

[6] R. Hicks and R. Bajcsy. Reflective surfaces as computational sensors. In *1999 Conference on Computer Vision and Pattern Recognition*, 1999.

[7] J. Holland. *Adaption in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Systems*. The University Press of Michigan, Ann Arbor, 1975.

[8] Y. Jin. A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing Journal*, 9(1):3–12, 2005.

[9] H. Kitano, Minoru Asada, Yasuo Kuniyoshi, Itsuki Noda, and Eiichi Osawa. RoboCup: The Robot World Cup Initiative. In *AGENTS '97: Proceedings of the first international conference on Autonomous agents*, pages 340–347. ACM Press, 1997.

[10] F. M. Marchese and D. G. Sorrenti. Omni-directional vision with a multi-part mirror. In P. Stone, T. Balch, and G. Kraetzschmar, editors, *RoboCup 2000: Robot Soccer World Cup IV*, volume 2019 of *Lecture Notes in Computer Science*, pages 179–188. Springer Verlag, 2001.

[11] C. F. Marques and P. U. Lima. A localization method for a soccer robot using a vision-based omni-directional sensor. In P. Stone, T. Balch, and G. Kraetzschmar, editors, *RoboCup 2000: Robot Soccer World Cup IV*, volume 2019 of *Lecture Notes in Computer Science*, pages 96–107. Springer Verlag, 2001.

[12] H.-P. Schwefel. *Numerical Optimization of Computer Models*. John Wiley & Sons, LTD, 1981.

[13] F. Streichert and H. Ulmer. JavaEvA - a java framework for evolutionary algorithms. Technical Report WSI-2005-06, Centre for Bioinformatics Tübingen, University of Tübingen, 2005.

[14] R. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *Robotics and Automation*, 3(4):323–344, August 1987.