Comparison of optimisation algorithms for deformable template matching

Vasileios Zografos

Linköping University, Computer Vision Laboratory
ISY, SE-581 83 Linköping, SWEDEN
zografos@isy.liu.se *

Abstract. In this work we examine in detail the use of optimisation algorithms on deformable template matching problems. We start with the examination of simple, direct-search methods and move on to more complicated evolutionary approaches. Our goal is twofold: first, evaluate a number of methods examined under different template matching settings and introduce the use of certain, novel evolutionary optimisation algorithms to computer vision, and second, explore and analyse any additional advantages of using a hybrid approach over existing methods. We show that in computer vision tasks, evolutionary strategies provide very good choices for optimisation. Our experiments have also indicated that we can improve the convergence speed and results of existing algorithms by using a hybrid approach.

1 Introduction

Computer vision tasks such as object recognition [1], template matching [2], registration [3], tracking [4] and classification [5] usually involve a very important optimisation stage where we seek to optimise some objective function, corresponding to matching between model and image features or bringing two images into agreement. This stage requires a good algorithm that is able to find the optimum value within some time limit (often in real-time) and within some short distance from the global solution.

Traditionally, such tasks have been tackled using local, deterministic algorithms, such as the simplex method [6], Gauss-Newton [7] or its extension by [8, 9] and other derivative-based methods [7]. Such algorithms, although usually improve on the solution relatively fast, need to be intialised near the proximity of the global optimum, otherwise they may get stuck inside local optima. In this work, we examine the simplex and the pattern search methods, due to their simplicity, ubiquity and tractability.

In recent years, a wide selection of global, stochastic optimisation algorithms have been introduced, the effectiveness of which, has ensured their use in computer vision applications. Their main advantage is that they are able to find

* This work has been carried out partially at University College London and at Linköping university under the DIPLECS project.

the optimum value without the need for good initialisation, but on the other hand require considerable parameter adjustment, which in some cases is not an intuitive or straightforward process. In addition they tend to be slow, since they require a higher number of function evaluations (NFEs).

This paper is organised as follows: in section 2 we present a selection of traditional local, algorithms, followed by the global approaches in section 3. In section 4 we explain our test methodology on the different datasets, including a set of 2-D test functions and real-image data of varying complexity. Section 5 includes an analysis of our experimental results for each algorithm, followed by an introduction to hybrid optimisation. We conclude with section 6.

2 Local methods

We consider two local optimisation methods that are well known and used in computer vision and many other scientific fields. These are the *downhill simplex* and the *pattern search* methods. A simplex is a polytope of $N+1$ vertices in N dimensions, which is allowed to take a series of steps, most notably the *reflection*, where the vertex with the worst function value is projected through the opposite face of the simplex to a hopefully better point. The simplex may also *expand* or *contract* or change its direction by *rotation* when no more improvements can be made. Simplex evaluations do not require calculation of function derivatives but the simplex must be initalised with $N+1$ points. This can be rather costly, but it still remains a very good solution when we need something working quickly for a problem with small computational overhead. We introduced two small yet

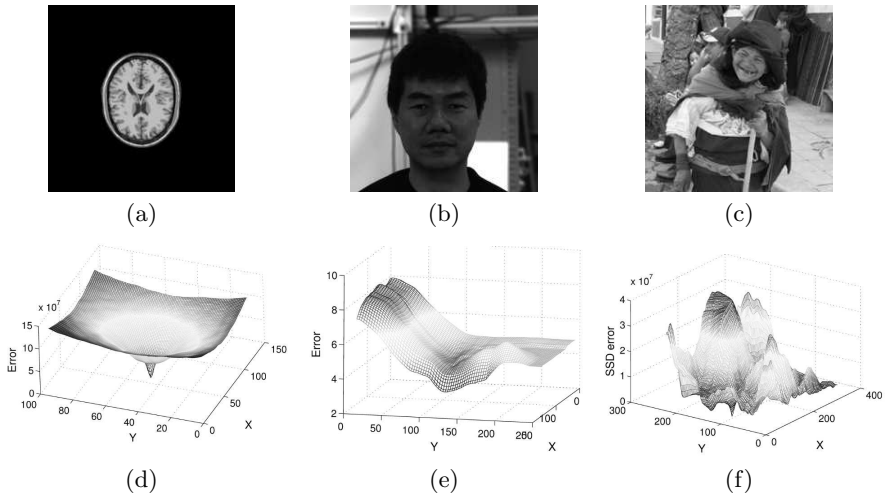

Fig. 1. Common test scenarios: Simple scene with constant background (a), moderate scene with background model available (b) and complex scene without background model (c). Their respective translation error surfaces are shown in (d), (e) and (f).

significant modifications to the basic algorithm [6], in order to deal with local minima. The first was the ability for the simplex to *restart* by generating N random unit vectors at distance λ from the current minimum, whenever its progress stalled. Furthermore, we gradually reduced the distance λ based on the number of function evaluations using a "cooling" schedule similar to Simulated Annealing [10].

Pattern search algorithms [11] conduct a series of exploratory moves around the current point, sampling the objective function in search of a new point (*trial point*) with a lower function value. The set of neighbourhood points sampled at every iteration is called a *mesh*, which is formed by adding the current point to a scalar multiple of a fixed set of vectors called the *pattern* and which itself is independent of the objective function. If the algorithm finds a new point in the mesh that has a lower function value than the current point, then the new point becomes the current point at the next step of the algorithm.

3 Global methods

In this section we introduce certain novel optimisation methods, specifically differential evolution and SOMA, together with a more traditional, generic Genetic Algorithm. Our aim is to determine whether or not stochastic, global algorithms are more effective in overcoming the typical convergence shortcomings associated with the aforementioned local methods, but also if these two new approaches are better suited than traditional genetic algorithms to computer vision problems.

A genetic algorithm (GA) [12] belongs to a particular class of optimisation methods based on the principles of evolutionary biology. Almost all GAs follow the basic stages of: *initialisation*, *selection*, *reproduction* (crossover, mutation) and *termination*. GAs have been applied to the solution of a variety of problems in computer vision, such as feature selection [13], face detection [14] and object recognition [15]. Additionally, they have been shown [16] to perform well in problems involving large search spaces due to their ability in locating good-enough solutions very early in the optimisation process.

Differential evolution (DE) [17] is an evolutionary population-based optimisation algorithm that is capable of handling non-differentiable, nonlinear and multi-modal objective functions, with any mixture of discrete, integer and continuous parameters. DE works by adding the weighted difference between two randomly chosen population vectors to a third vector and the fitness result is

	Simplex	P. Search	GA	DE	SOMA
Sphere	**26**, 3.09E-5	81, **0**	4600, 1.62E-5	1600, 6.44E-5	1302, 9.5E-5
Rosenbrock	**70**, 8.09E-5	89, **0**	10^4, 1.1E-2	2800, 6.59E-5	10^4, 1.34E-2
Griewank's	10^3, 7.93E-3	10^3, 7.39E-3	8300, **4.85E-5**	**2100**, 9.24E-5	10^4, 7.4E-3
Rastrigin's	516, 2.28E-5	**81**, **0**	10^4, 6.45E-4	2300, 9.92E-5	4570, 1.57E-5
Camel-back	**30**, 3.0E-5	169, 3.0E-5	10^4, 4.9E-4	1900, 1.0E-4	2651, **0**

Table 1. Comparative results from the 2-D test functions using all the algorithms.

compared with an individual from the current population. In this way, no separate probability distribution is required for the perturbation step and DE is completely self-organising. DE has been used successfully in a variety of engineering tasks.

Finally, we examine the Self-Organizing Migrating Algorithm or SOMA, a stochastic optimization algorithm that is modelled on the social behaviour of co-operating intelligent individuals and was chosen because of its proven ability to converge towards the global optimum [18]. SOMA maintains a population of candidate solutions. In every iteration, the whole population is evaluated and the individual with the highest fitness (or lower error value) is designated as the leader. The remaining individuals will "migrate" towards the leader, that is, travel in the solution space at the direction of the fittest individual.

4 Experimental domain

Our task now is to compare these different strategies against a set of 2-D, analytic functions and real-image data. The aim is to determine the general properties of each of the optimisation algorithms and understand some details about their parameter settings. We may then use this information and apply the same algorithms in a template matching problem and see how they compare in more realistic circumstances.

4.1 2-D test functions

These functions are designed to test against universal properties of optimisation algorithms and give us an overall understanding of each method's strengths and weaknesses and possible parameter choices, before moving on to template matching specific datasets and experimentation. The original inspiration was the work of [19], but with a few modifications, which include: the *sphere model*, $f(x_i) = \sum x_i^2$, a smooth, unimodal, symmetric, convex function used to measure the general efficiency of an optimisation algorithm.

Rosenbrock's function, $f(x_i) = \sum[(1 - x_i)^2 + 100(x_{i+1} - x_i^2)^2]$, which has a single global minimum inside a long, parabolic-shaped flat valley. Algorithms that are not able to discover good directions underperform in this problem by oscillating around the minimum.

The *six-hump camel-back* function, $f(x, y) = (4 - 2.1x^2 + \frac{x^4}{3})x^2 + xy + (-4 + 4y^2)y^2$ which has a wide and approximately flat plateau and a number of local minima. In addition, it has two, equally important global minima. Unless an algorithm is equipped to handle variable step sizes, then it is likely to get stuck in one of the flat regions.

Rastrigin's function $f(x_i) = 10n + \sum (x_i^2 - 10\cos(2\pi x_i))$ and the slightly more difficult *Griewank's* function $f(x) = \sum \frac{x_i^2}{4000} - \prod \cos(\frac{x_i}{\sqrt{i}}) + 1$. Both have a cosine modulation part that simulates the effects of noise (multiple modes), and are designed to test whether an algorithm can consistently jump out of local minima.

4.2 Real-image template matching

In this section we propose more detailed experiments relevant to computer vision by examining deformable template matching, since it is a generic scenario that might be applied to many different areas in the field. Template matching can be expressed as the task of searching for the parameters ξ of a transformation T that will bring the model template F_0 into agreement with an image I. The transformation T, for 2-D problems, is usually an affine transformation with 6 parameters. In mathematical terms this is defined as a minimisation problem:

$$\min_{\xi} S = \sum\nolimits_{x,y} g(I(x,y), TF_0(x,y)), \tag{1}$$

where $g(.,.)$ is some dissimilarity metric and the sum is over all the features in the template, in this case pixels. When g is chosen as the sum of squared differences (SSD) dissimilarity metric, (1) produces specific error surfaces which have been examined in previous work by [20] with well known properties. For ease of analysis and visualisation we consider the transformation parameters as independent with the following parameterisation: $T = SRU_x + D$, that is a 2-D translation D, anisotropic scaling S and 1-D rotation R and shear U_x.

Of particular interest to us is the translation transform, because it contains the majority of problems for optimisation algorithms. This is due to the fact that, in general, a change in translation will move the model away from the object and on to the background region where unknown data and thus more noisy peaks in the error surface exist. Furthermore, the translation surface may vary depending on the type of template model F_0 and scene image I we use. For example, if we consider the object of interest in front of a constant background (see Fig. 1(a)), then the translation space (assuming all other transformation parameters are optimally set) is a simple convex surface (Fig. 1(d)). This is considered to be a relatively easy scenario of a computer vision optimisation problem and it is mostly encountered in controlled environments (e.g. assembly line visual inspection, medical image registration and so on).

A second possibility, is for the scene image I background to be substantially more complex (see Fig. 1(b)) with non-trivial structure and noise existent. In this case, our template model F_0 may be more elaborate also, composed of a full foreground and background model. As such, we either have to know what the background is [21], build a very simple model [20], or have a statistical model of what it is expected to be like [22]. The result will be a translation error surface as in (Fig. 1(e)), which constitutes a moderate optimisation problem, with most global algorithms and a number of local methods under good initialisation, expected to converge to the correct minimum.

Finally, we have the hardest case, where considerable structure and noise exist in the scene image background, but a model of the latter is not available (see Fig. 1(c)). The optimisation difficulty in this scenario is apparent in the complexity of the produced 2-D translation error surface (Fig. 1(f)), and all local optimisation methods not initialised in close proximity to the global minimum are expected to fail, while most global methods will converge with great difficulty and after many iterations.

Regarding the remaining error spaces, we would like to draw attention to the irregularities of the 2-D scale space previously examined in [23]. Finally, the rotation and shear spaces can be easily minimised, even though for the rotation space there may be a number of local minima at angle intervals of $\pm\pi/2$, depending on the rotation symmetrical properties of the object.

5 Experiments: methods and results

We now present the experimental methods for each dataset, the algorithm configurations and the comparative results from which we aim to draw certain conclusions about the fitness and efficiency of each strategy in relation to the typical computer vision scenario.

5.1 Set 1: 2-D test functions

We used the total NFEs as a general and independent, quantitative measurement for comparing different algorithms. Convergence was defined as a recovered error minimum no greater than $\tau = 10^{-4}$ of the known global solution, and inside the allocated optimisation budget (1000 NFEs for local methods and 10000 NFEs for global methods). Additionally, we tried to use similar initialisation criteria for every method, in order to make intra-category comparison easier. For stochastic approaches, we carried out 5 test runs per function and averaged the results.

The initialisation settings for each method where: **Simplex** initial triangle [(5,5),(5,0),(-5,-5)]; **Pattern Search** starting point (4,5) and polling of the mesh points at each iteration using the *positivebasis2N* [11] method; **GA** population generated from U([-5,5]) and using a *stochastic uniform* selection and *scattered cross-over* reproduction functions [12]; **DE** population limit=100, maximum iterations=100, F=0.8, and CR=0.5 [17]. *The Best1Bin* strategy was also chosen and the algorithm was initialised inside the soft boundaries [-5,5]; Finally, for **SOMA** step=0.11, pathLength=2, prt=0.1, migrations=50 and populationsize=10, which approximates to 10000NFEs. The best strategy, was the *all-to-one* randomly [18] and the algorithm was initialised inside the hard boundary [-5,5]. The combined output for all methods is shown in Table 1. The first number in each column corresponds to the average NFEs for this method, while the second is the absolute difference between the global and recovered minima. The bold figures represent the best performing algorithm for each function.

As we can see, the Simplex performs rather well with very low required NFEs. It can also cope well with flat region uncertainties due to its expanding/contracting nature, and negotiate moderately noisy surfaces (Rastrigin's) albeit with a high NFEs. This is not the case however when numerous local minima exist (Griewank's) even if the available NFEs are increased. The pattern search method requires more NFEs than the Simplex indicating that it is not so efficient nor can it discover good directions. It did however find the exact location of the global minimum in most cases and managed to deal with moderate noise much more efficiently than the Simplex. Regarding flat regions, its fixed

mesh expansion and contraction factors were not very adequate in cases where there was no information about the current function estimate. The GA is the worst algorithm and fails to converge below the 10^{-4} threshold for NFEs=10000. Nevertheless, it can cope well with noise the majority of times and thus it is best suited for difficult problems with inexpensive function cost where a high NFEs would be justified. DE is the best across most functions and is generally more efficient than both GA and SOMA. It does also succeed in solving Griewank's function (80% of times), which as we have already seen is particularly problematic for all optimisaiton algorithms so far. Finally, SOMA performs somewhere between GA and DE, is quite efficient for simple test functions and can deal with a moderate amount of noise (not Griewank's function though), also allowing for flat surface uncertainty with a varying step length when improvement stalls. However, it is not good in determining good search directions since it was not able to converge in the Rosenbrock's function, although it did come close.

What these tests have demonstrated is that the reducing-step restarting Simplex and the DE algorithms were the best performing from the local and global methods respectively. Before we can draw any broader conclusions however, we need to perform more rigorous tests on real-image data.

5.2 Set 2: Real-image template matching

We shall further analyse the fitness of each of the examined optimisation algorithms by performing more detailed tests with the 3 real-image datasets previously seen (easy, moderate and hard) using the objective function in (1). We define convergence in this context as the ability to recover a model configuration within some Euclidean distance threshold from the known optimum. We could have also used the minimum value of (1) to determine convergence, but in this case and especially when using a SSD dissimilarity metric, it is quite possible to find an invalid model configuration with an error value that is lower or equal to the global minimum, as discussed in [23]. As such, the threshold boundaries were defined as follows: translation $t_x, t_y = 5$, scale $s_x, s_y = 0.1$, rotation $\theta = 10^0$, and shear $\phi = 5^0$. Any configuration within these limits from the known global

		Simplex	P. Search	GA	DE	SOMA
	Convergence %	2	12	0	**100**	**100**
Dataset 1	NFEs	1060	**476**	-	3915	2551
	Minimum	2.945	1.365	-	**0.3213**	0.3265
	Convergence %	2	3	11	**96**	61
Dataset 2	NFEs	476	0*	**446**	889	1416
	Minimum	0.09	0.0915	0.08815	**0.0799**	0.0865
	Convergence %	1	4	63	61	**97**
Dataset 3	NFEs	1194	**862**	4603	11483	4070
	Minimum	0.03806	0.0389	0.0273	0.0301	**0.0252**

Table 2. Comparative results from the 3 datasets using all the algorithms.

minimum will be considered a valid solution and convergence will be deemed as successful. The same values have been used across all the 3 datasets. We can now define a number of quantitative measures such as the *global minimum* of a *converged* test run; the *time to convergence*, that is how many iterations before the optimisation reached the convergence thresholds and the *convergence percentage*, that is the number of times the optimisation converged inside the set threshold.

In all the following tests, we used a maximum of 2000 and 20000 NFEs for the local and global optimisation methods respectively, and each method was allowed to perform 100 separate tests, the results of which were averaged. None of the algorithms was initialised close to the ground truth solution, but instead and in order to eliminate any bias, they were started randomly within the 6 coefficient domains. In more detail, we used the following settings for each method: **Simplex** algorithm with random initial 7×6 simplex within the boundaries [1-50, 1-50, 0.5-1, 0.5-1, 1-20, 1-20], step size λ=[20, 20, 2, 2, 50, 20] and cooling rate r=0.9; **Pattern search** initial randomly generated population in the range [0-100, 0-100, 0.5-1.5, 0.5-1.5, 0-50, 0-10]. Poll method = *positiveBasis2N*, initial mesh size=30, rotate and scale mesh, expansion factor=2 and contraction factor=0.5; **GA** 200 generations and 100 populations. Initial population function $U([0-100],[0-100],[0-1],[0-1],[0-50],[0-10])$; **DE** populations=100, maximum iterations=200, F=0.8, CR=0.5, strategy=*Best1Bin*. Soft boundaries=[1-100, 1-100, 0.5-2, 0.5-2, 0-100, 0-50]. Finally **SOMA** step=0.5, pathLength=1.5, prt=0.1, migrations=100, popsize=50. Hard boundaries=[1-100, 1-100, 0.5-2, 0.5-2, (-180)-180, (-50)-50]. These settings were kept fixed throughout all the datasets.

Dataset 1 - MRI image: The first test data consist of an MRI scan of a human brain in front of a black background (Fig. 1(a)). A template of the object was subjected to a 2-D affine transform with: $(t_x, t_y) = 65, 68$; $(s_x, s_y) = 0.925, 1.078$; $\theta = -25$ and $\phi = -5.5826$. The SSD error between the optimal template and the scene, including minor interpolation and approximation errors, is 6.6689. After 100 experimental runs with each algorithm, we obtained the results in rows 2-4 of Table 2.

It is clear that both DE and SOMA have the best performance, with all their test runs converging inside the threshold. DE uses only about 20% of its optimisation budget to achieve convergence on average, but SOMA is the clear winner with approximately 1400 less NFEs required for comparable results. Next we have the genetic algorithm which very suprisingly did not manage to converge

	Dataset 1	Dataset 2	Dataset 3
Convergence % (±%)	86% (-14%)	41% (-33%)	81% (-16.5%)
Hybrid SSD @ 6000 NFEs (±%)	0.4275 (-65%)	0.0868 (-24%)	0.02661 (-22%)
SOMA SSD @ 6000 NFEs	1.215	0.1138	0.03419
SOMA SSD @ 20000 NFEs (±%)	0.3265 (−73%)	0.08659 (−24%)	0.02523 (−26%)

Table 3. The results of the hybrid and SOMA tests at 6000 and 20000 NFEs.

in any of the 100 tests but instead converged inside one of the many pronounced local minima of the rotation parameter θ (due to the symmetry of the human brain scan), while having successfully identified the other parameters.

From the local methods, due to the absence of good initialisation, we expect much lower convergence rates than the global methods. Compared between themselves, the pattern search search can converge many more times and at around half the NFEs than the simplex requires.

Dataset 2 - CMU PIE data: The second installment of tests was carried out in a real image sample (see Fig. 1(b)) from the CMU PIE database [21], with a complex background but which is given as a separate image. This is a more difficult scenario than previously and we expect a lower convergence rate across all the methods. In this occasion, the ground truth is located at $[82, 52, 1.0786, 1.1475, 10^0, -4.8991^0]$ with an SSD error of 0.1885. After 100 test runs for each optimisation algorithm, we obtain the results in rows 5-7 of Table 2. As expected, we see an overall drop in the recognition results with DE being the best performing method, while at the same time displaying convergence behaviour reminiscent of a local method; that is, converging in under 900 NFEs. The rest of the algorithms perform rather poorly, with SOMA at 61% and GA at a much lower 11%. Furthermore, all methods find a good minimum at <0.1, which is lower than the known solution, since they can effectively overcome any inherent interpolation and approximation errors. We also note that in the case of the pattern search algorithm, the only 3 cases that succeeded in converging correctly, were the ones that randomly initialised inside the basin of attraction.

Dataset 3 - Real image data without a background model: Finally we arrive to the hardest case; A real image with a complex background, but without any model of the latter (see Fig. 1(c) and (f)). Due to the increased difficulty associated with this particular dataset, it is expected that the overall optimisation performance will be further reduced. The optimal SSD solution in this case is $[106, 59, 0.9048, 1.0444, 12.02^0, 0^0] = 0.0488$. If we use the same optimisation settings as we did previously, we get the following results after 100 test runs (Table 2 rows 8-10). SOMA performs very well with a 97% convergence ratio, with the GA coming second at 63% and DE not particularly efficient at 61%. We also see that it takes DE many more iterations in order to converge, whereas SOMA and GA on average reach the global minimum around 2.5 times faster. Despite that, all the global methods reached approximately the same minimum error. Both the local methods managed to converge fast and towards a very good solution but for only a limited number of cases, most probably due to the absence of good starting points.

In conclusion we may say that both DE and SOMA perform consistently well in all the 3 cases, with an expected performance penalty associated with the increased difficulty of each dataset, and both reach approximately the same minimum at the end of their allocated time budget. Where they differ however, is in the time they require for initial convergence, with SOMA being the clear winner since it manages to approximate the correct solution much earlier than DE. This makes SOMA ideal for the hybrid approach later on. As far as the GA

is concerned, we have seen that it can reach an equally good minimum error, just like SOMA and DE, when and if it converges successfully. Nevertheless, it has the tendency to get stuck in pronounced local minima in all but the simplest datasets, which consequently reduces its effectiveness on template matching-based object recognition. The two local methods, simplex and pattern search, can converge very fast and nearly at the same minimum whenever they can reach its proximity. We can therefore use either one for the hybrid approach next.

5.3 Hybrid approach

The hybrid approach is essentially the combination of a global, stochastic algorithm (in this case SOMA) designed to get us close to the basin of attraction as early as possible from a random, distant location on the error surface, and a local method (Simplex), whose purpose is to rapidly refine the good recovered solution, much faster and more efficiently than the global method alone can. The only additional issue with using a hybrid method, is how to determine when is the best time to switch between methods. One possibility, is to use a number of concurrent criteria to decide when we are close to the switch point. The first such criterion, could be a proximity threshold such as the Euclidean distance previously used to determine convergence. Another could be the observed relative gain of each successful iteration. When the gain is below some predetermined value, we can assume that the global algorithm has near-stalled and switch to the local method. Finally, a third criterion might be the relative change of each parameter at every iteration. Alternatively, we may opt to use a fixed NFE-related threshold, based on the information we have about the optimisation behaviour of SOMA. If for example we revisit Table 2 we can see that on average and across all 3 datasets, SOMA requires between 1500-4000 NFEs to reach the minimum error threshold. We can therefore use this prior knowledge and set SOMA to run at a fixed number of 4000 NFEs. Such a number will most likely ensure that the simplex switch is performed when we are near the solution.

The only settings that we altered since the previous test runs are: **SOMA** migrations=20, popsize=50 $\approx$ 4000 NFEs; **Simplex** 2000 NFEs, initial 7×6 simplex that includes as a vertex V_1 the optimum recovered solution from the

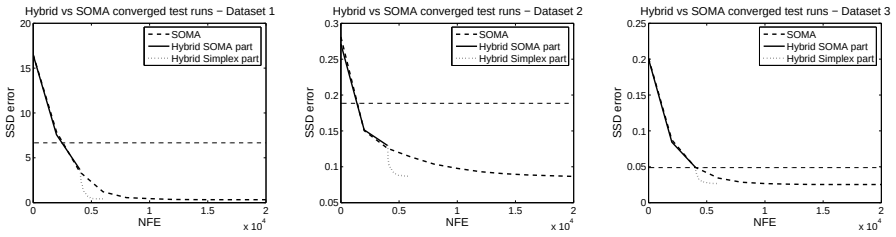

Fig. 2. Plots comparing the hybrid approach and the SOMA method for the 3 datasets.

SOMA run and 6 random vertices V_{2-7} generated at a distance d=[5,5,0.1,10,5] from V_1. Note, this is the Euclidean distance threshold from earlier. We carried out 100 test runs of the hybrid method for each of the 3 datasets and we present the results on Table 3. The second row shows the convergence rate of the hybrid method. The percentage differences (±%) in this row are in relation to the original SOMA results (column 7 of Table 2). The next two rows show the average SSD error of the 100 hybrid runs and the original 100 SOMA runs at 6000 NFEs. The percentage differences of row 3 are in relation to the results in row 4. Finally, the last row shows the average SSD error of the original 100 SOMA runs at the maximum 20000 NFEs, with a percentage difference in relation to the results in row 4. We see that the convergence ratio of the hybrid, is only around 15-30% lower than the original tests, but the error is between 20-65% lower than the SOMA-only approach for the same NFEs. In fact, the error values are quite close to the original recovered minima using the full 20000 NFEs. This can also be seen in the iteration plots in Fig. 2, where we can observe the secondary drop of the local method, which always manages to refine the optimisation further (i.e. there is no stall at the switch point), indicating that on average we have chosen good switch points and that the local method can converge faster that the global method for the same number of iterations.

We may therefore say that by using a hybrid approach, it is possible to obtain results that are very close to a global algorithm-only solution, but at a considerably reduced NFEs cost. In that sense, a hybrid optimiser might be useful in situations where we are faced with a costly objective function but a good initialisation for a local-only method is not available. With the application of the hybrid method we may still use a global algorithm for initialisation, while avoiding the increased NFEs overhead.

6 Conclusion

In this paper we have examined the suitability of a number of different optimisation methods (both novel and traditional) for the task of template matching. We have tested against a series of 2-D, analytic functions designed to highlight the generic properties of each optimisation method, followed by three realistic datasets of progressive difficulty, commonly encountered in computer vision. Our results show that the novel methods outperform the traditional approaches in all cases, and we hope that this work serves as a first step into introducing these novel methods to the computer vision community and establishing them as better alternatives to the methods currently being used. Finally we argue that a hybrid combination of a global and local methods can produce equally good results in a fraction of the time required by a global method alone. We demonstrate this to some degree, with a number of additional experiments.

References

1. Peters, G.: Theories of three-dimensional object perception: A survey,. Recent Research Developments in Pattern Recognition, Transworld ResearchNetwork (2000)

2. Jain, A.K., Zhong, Y., Dubuisson-Jolly, M.P.: Deformable template models: A review. Signal Processing **71** (1998) 109–129
3. Hill, D.L.G., Batchelor, P.G., Holden, M., Hawkes, D.J.: Medical image registration [invited topical review]. Physics in Medicine and Biology **46** (2001) R1–R45
4. Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. ACM Comput. Surv. **38** (2006) 1–45
5. Hasegawa, O., Kanade, T.: Type classification, color estimation, and specific target detection of moving targets on public streets. Machine Vision and Applications **16** (2005) 116–121
6. Nelder, J.A., Mead, R.: A simplex method for function minimization. Computer Journal **7** (1965) 308–313
7. Nocedal, J., Wright, S.: Numerical optimization. Springer, New York (1999)
8. Levenberg, K.: A method for the solution of certain problems in least squares. Quart. Appl. Math. **2** (1944) 164–168
9. Marquardt, D.: An algorithm for least-squares estimation of nonlinear parameters. SIAM J. Appl. Math. **11** (1963) 431–441
10. Betke, M., Makris, N.C.: Fast object recognition in noisy images using Simulated Annealing. In: Proceedings of the 5th ICCV. (1995) 523–530
11. Audet, C., Jr., J.E.D.: Analysis of generalized pattern searches. SIAM J. on Optim. **13** (2003) 889–903
12. Holland, J.H.: Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. The MIT Press (1992)
13. Kim, H.D., Park, C.H., Yang, H.C., Sim, K.B.: Genetic algorithm based feature selection method development for pattern recognition. In: International Joint Conference SICE-ICASE. (2006)
14. Bebis, G., Uthiram, S., Georgiopoulos, M.: Genetic search for face detection and verification. In: ICIIS. (1999) 360–367
15. Hill, A., Taylor, C.J., Cootes, T.F.: Object recognition by flexible template matching using genetic algorithms. In: Proceedings of the 2nd ECCV, London, UK, Springer-Verlag (1992) 852–856
16. Goldberg, D.E.: Genetic Algorithms in Search, Optimzation & Machine Learning. Addison-Wesley (1989)
17. Storn, R., Price, K.V.: Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. Journal of Gl. Optim. **11** (1997) 341–359
18. Zelinka, I.: SOMA-Self Organizing Migrating Algorithm. In Onwubolu, G., Babu, B.V., eds.: New optimization techniques in engineering. Springer, Berlin (2004)
19. DeJong, K.A.: An analysis of the behavior of a class of genetic adaptive systems. PhD thesis, University of Michigan, Ann Arbor, MI, USA (1975)
20. Buxton, B., Zografos, V.: Flexible template and model matching using image intensity. In: Proceedings DICTA. (2005) 438 – 447
21. Sim, T., Baker, S., Bsat, M.: The CMU pose, illumination and expression (PIE) database. In: Proc. of the 5th FG. (2002)
22. Srivastava, A., Lee, A., Simoncelli, E., Zhu, S.C.: On advances in statistical modeling of natural images. Journal of Math. Imag. and Vis. **18** (2003) 17–33
23. Zografos, V., Buxton, B.F.: Affine invariant, model-based object recognition using robust metrics and Bayesian statistics. In: Proc. of the ICIAR 2005. Volume 3656. (2005) 407–414