19

Registration of Point Patterns Using Modern Evolutionary Algorithms

Peng-Yeng Yin
Department of Information Management, National Chi Nan University
Taiwan

1. Introduction

Registration of point patterns is a fundamental process prior to many applications such as image alignment, object recognition, and pattern retrieval. When two images are aligned, people prefer to deal with sets of local features (for example, dominant points) instead of pixel arrays to increase the accuracy and save the computational time. Given two point patterns, the aim of point pattern registration (PPR) problem is to find an optimal geometric transformation which transforms one point pattern by reference to the other such that a dissimilarity measure between them is minimized.

PPRs can be classified into various categories according to two features, *completeness* and *label*. Complete registration stipulates that the two registered patterns should have exactly the same number of points and there exists a one-to-one correspondence mapping between the members of the two point sets. While the incomplete registration deals with patterns with missing and spurious points, a mapping between subsets of the point patterns is thus sought. On the other hand, labeled registration is conducted using the *a priori* information (e.g., point order, intensity, gradient, etc.) as well as the point coordinates. While unlabeled registration determines the point correspondences based on the coordinates information of the data points only. Conspicuously, incomplete unlabeled registration is the hardest category of all the PPR classifications.

The PPR considered in this chapter is confined by the affine transformation consisting of rotation, scaling, and translation. Let $A = \{a_i \mid i = 1, 2, ..., n\}$ and $B = \{b_i \mid i = 1, 2, ..., m\}$ be two point patterns in R^2 and they are affinely-dependent under a transformation $T = (\theta, s, t_x, t_y)$ where θ denotes the rotation angle, s is the scale factor, and t_x and t_y are the translation offsets along the directions of x- and y-axis. Also let (a_i, b_j) be one of the pair-wise point mappings with T, and denote by $a_i = [x_{a_i} \quad y_{a_i}]^T$ and $b_j = [x_{b_j} \quad y_{b_j}]^T$ the corresponding coordinates, we have the following affine relation.

$$T(a_i) = s\begin{bmatrix}\cos\theta & -\sin\theta \\ \sin\theta & \cos\theta\end{bmatrix}\begin{bmatrix}x_{a_i} \\ y_{a_i}\end{bmatrix} + \begin{bmatrix}t_x \\ t_y\end{bmatrix} = \begin{bmatrix}x_{b_j} \\ y_{b_j}\end{bmatrix} = b_j \tag{1}$$

Nevertheless, the ideal transformation (1) usually does not hold under many real situations such as the existence of missing and spurious points and the distortion of patterns, resulting

Source: Pattern Recognition Techniques, Technology and Applications, Book edited by: Peng-Yeng Yin,
ISBN 978-953-7619-24-4, pp. 626, November 2008, I-Tech, Vienna, Austria

in a registration error. Two error dissimilarity measures between two aligning point patterns are broadly used in the literature to assess the quality of the registration result.

- *Agrawal's Heuristic Dissimilarity* (AHD) *Measure* Let Ω be the set of point correspondences between the two patterns with $|\Omega| \le \min(n, m)$. The registration error of Ω with respect to transformation *T* can be evaluated using the integral squared error defined as

$$\varepsilon^2 = \sum_{(a_i, b_j) \in \Omega} \left\| T(a_i) - b_j \right\|^2 \tag{2}$$

where $\|\bullet\|$ indicates the vector length in the Euclidean space. Agrawal *et al.* (1994) proposed an overall registration dissimilarity measure as

$$AHD(A, B) = \begin{cases} \dfrac{\varepsilon^2}{s|\Omega|}\left(1 + \left(\dfrac{m-2}{|\Omega|-2}\right)\log_2\left(\dfrac{m-2}{|\Omega|-2}\right)\right) & |\Omega| \ge 3 \\ \infty & |\Omega| = 0,\ 1,\ 2 \end{cases} \tag{3}$$

The AHD measure is normalized with the scale factor *s* and includes a penalty term for the unregistered points in the searched pattern.

- *Partial Hausdorff Distance* (PHD) Huttenlocher *et al.* (1993) used the directed partial Hausdorff distance from *A* to *B* as

$$DPHD_k(A, B) = K^{th}_{a_i \in A} \min_{b_j \in B} \left\| T(a_i) - b_j \right\| \tag{4}$$

where K^{th} returns the *k*th smallest value of $\min_{b_j \in B} \left\| T(a_i) - b_j \right\|$ for all $a_i \in A$. The directed partial Hausdorff distance from *B* to *A* can be analogously defined as

$$DPHD_k(B, A) = K^{th}_{b_j \in B} \min_{a_i \in A} \left\| b_j - T(a_i) \right\| \tag{5}$$

Finally, the partial Hausdorff distance from both patterns is given by

$$PHD_k(A, B) = \max\left(DPHD_k(A, B),\ DPHD_k(B, A)\right) \tag{6}$$

In contrast to AHD, the PHD measure only takes into account the registered points such that the situation of incomplete registration can be accommodated.

Many approaches have been proposed for tackling various PPR problems. According to our recent survey, there was a departure in the PPR approaches in 1990s. Traditional approaches take advantage of the geometric properties involved with the point patterns to improve the search efficiency and effectiveness. More recently, some evolutionary algorithms were proposed to evolve the optimal transformation between the given point sets. The conceptions of existing methods are summarized as follows.

- *Clustering* The technique (Chang et al., 1997; Goshtasby & Stockman, 1985; Umeyama, 1991; Wang & Chen, 1997; Yuen, 1993) calculates the registration transformation parameters θ, *s*, t_x and t_y for each pair of points contained in both patterns and increases the frequency count of the corresponding cell (θ, *s*, t_x, t_y) in an accumulator. The clusters

of the cells with respect to the frequencies are detected. The peak of the cluster with the maximum frequency corresponds to the optimal transformation parameters. Clustering methods are computationally intensive due to the large number of combinations of point pairs and the dimensionality of the parameter space.

- *Parameter decomposition* The method (Griffin & Alexopoulos, 1991; Huttenlocher et al., 1993; Olson & Huttenlocher, 1997) divides the parameter estimation process into multiple phases. At the first phase, a selected parameter is estimated based on the domain knowledge such as the geometric invariant constraints. Then, at each of the following phases, one or more of the remaining parameters are estimated by reference to the partial parameters values previously determined, hence, the number of possible combinations between values of separate parameters is greatly reduced. However, the inaccuracy of parameter estimation could be magnified due to successive propagation through various phases.
- *Relaxation* The technique (Ogawa, 1984; Ranade & Rosenfeld, 1980; Ton and Jain, 1989) iteratively updates the merit score of every point mapping (a_i, b_j) from both patterns given the merit scores of the other interacting point mappings. The interacting point mappings are those that are mutually constrained for registration due to geometry properties. The algorithm converges when those merit values become consistent (or hardly changed) between consecutive iterations and the point mappings with the maximum merits are considered as the true transformation point correspondence.
- *Bounded alignment* Mount *et al.* (1999) proposed a geometric branch-and-bound search of the transformation space and used the point alignment information to bound the search. They specify an approximation factor to guarantee the accuracy of the final match and use point alignments when a significant number of point correspondences can be inferred to accelerate the search. The robustness of the algorithm has been demonstrated on registration of real satellite images.
- *Spectral graph analysis* Carcassoni and Hancock (2003) applied the spectral graph theory to compute the point correspondence. The global structural properties of the point pattern are ascribed by the eigenvalues and eigenvectors of the proximity weighting matrix. The influence of the contamination and drop-out in the point pattern is discounted via the EM algorithm so the accuracy of the registration is increased.
- *Genetic algorithms* Some researchers (Ansari et al., 1990; Zhang et al., 2003) have applied genetic algorithms to explore the search space of point mappings. The chromosomes encode instances of point mappings and evolve by performing genetic operators to reduce the AHD or PHD dissimilarity values, such that the optimal registration transformation can be obtained.
- *Simulated annealing* The authors of (Ansari et al., 1993; Starink & Backer, 1995) employed the simulated annealing technique to tackle the PPR problem. The identification of point correspondences between two point patterns is mathematically formulized as energy minimization. The registration error incurred by the current configuration of point correspondences is treated as the energy of that configuration. Simulated annealing rearranges the particles of configuration to reach thermal equilibrium at various temperature levels and finally converges to an optimum configuration as the system is frozen.

This chapter investigates the strengths and weaknesses of applying modern evolutionary algorithms, in particular, the particle swarm optimization and scatter search, to cope with

the incomplete unlabeled PPR problem. The performance of the two algorithms is evaluated by competing with existing algorithms on synthetic datasets. The experimental results manifest that the modern evolutionary algorithms are superior and malleable against varying scenarios such as positional perturbations, contaminations and drop-outs from the point patterns.

The remainder of this chapter is organized as follows. Section 2 reviews the underlying modern evolutionary algorithms. Section 3 presents the proposed methods for the PPR problem. In Section 4, the experimental results are illustrated. Finally, a conclusion is given in Section 5.

2. Modern evolutionary algorithms

The notion of evolutionary algorithms has been introduced since 1960's and usually refers to a class of genome-inspired computation algorithms consisting of genetic algorithms, evolutionary programming, evolutionary strategy and genetic programming. These novel algorithms have exhibited great successes in many engineering and science applications. In the mid 1990's, another class of evolutionary algorithms emerged. These algorithms are bio-inspired and established on metaphors of socio-cognition. Typical examples in this class include culture algorithms, ant colony optimization, particle swarm optimization and scatter search. This chapter is focused on the application of particle swarm optimization and scatter search to the point pattern registration problem. In this section, we give a brief review of the two modern evolutionary algorithms.

2.1 Particle swarm optimization

Particle swarm optimization (PSO) is a new evolutionary algorithm proposed in (Kennedy & Eberhart, 1995). PSO is bio-inspired and it models the social dynamics of bird flocking. A large number of birds flock synchronously, change direction suddenly, scatter and regroup iteratively, and finally perch on a target. This form of social intelligence not only increases the success rate for food foraging but also expedites the process. The PSO algorithm facilitates simple rules simulating bird flocking and serves as an optimizer for continuous nonlinear functions. The general principles of the PSO algorithm can be outlined in the following features.

- *Particle representation* The particle in the PSO is a candidate solution to the underlying problem and move iteratively and objectively in the solution space. The particle is represented as a real-valued vector rendering an instance of all parameters that characterize the optimization problem. We denote the ith particle by $P_i = (p_{i1}, p_{i2}, ..., p_{id})^T \in R^d$, where d is the number of parameters.
- *Swarm* The PSO explores the solution space by flying a number of particles, called swarm. The initial swarm is generated at random and the size of swarm is usually kept constant through iterations. At each iteration, the swarm of particles search for target optimal solution by referring to previous experiences.
- *Personal best experience and swarm's best experience* The PSO enriches the swarm intelligence by storing the best positions visited so far by every particle. In particular, particle i remembers the best position among those it has visited, referred to as $pbest_i$, and the best position by its neighbors. There are two versions for keeping the neighbors' best position, namely *lbest* and *gbest*. In the local version, each particle keeps

track of the best position *lbest* attained by its local neighboring particles. For the global version, the best position *gbest* is determined by any particles in the entire swarm. Hence, the *gbest* model is a special case of the *lbest* model. It has been shown that the local version is often better, particularly the one using random topology neighborhood where each particle generates L links at random after each iteration if there has been no improvement i.e. if the best solution seen so far by the swarm is still the same. In our implementation, we set $L = 10$.

- *Particle movement* The PSO is an iterative algorithm according to which a swarm of particles fly in the solution space until the stopping criterion is satisfied. At each iteration, particle i adjusts its velocity v_{ij} and position p_{ij} through each dimension j by reference to, with random multipliers, the personal best position ($pbest_{ij}$) and the swarm's best position ($lbest_{ij}$, if the local version is adopted) using Eqs. (7) and (8) as follows.

$$v_{ij} = K[v_{ij} + c_1 r_1(pbest_{ij} - p_{ij}) + c_2 r_2(lbest_{ij} - p_{ij})] \tag{7}$$

and

$$p_{ij} = p_{ij} + v_{ij} \tag{8}$$

where c_1 and c_2 are the cognitive coefficients and r_1 and r_2 are random real numbers drawn from $U(0, 1)$. Thus the particle flies toward *pbest* and *lbest* in a navigated way while still exploring new areas by the stochastic mechanism to escape from local optima. Clerc & Kennedy (2002) has pointed out that the use of the constriction factor K is needed to insure convergence of the algorithm and its value is determined by

$$K = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|} \tag{9}$$

where $\varphi = c_1 + c_2$, $\varphi > 4$. Typically, φ is set to 4.1 and K is thus 0.729.

- *Stopping criterion* The PSO algorithm is terminated with a maximal number of iterations or the best particle position of the entire swarm cannot be improved further after a sufficiently large number of iterations.

PSO has received great successes in many applications including evolving weights and structure for artificial neural networks (Eberhart & Shi, 1998), manufacture end milling [21], state estimation for electric power distribution systems (Shigenori et al., 2003), and curve segmentation (Yin, 2004). The convergence and parameterization aspects of the PSO have been also discussed (Clerc & Kennedy, 2002; Trelea, 2003).

2.2 Scatter search

Scatter search (SS) is another new evolutionary algorithm proposed in (Glover, 1998), although its original proposal may appear in an earlier literature (Glover, 1977). SS operates on a set of diverse elite solutions, referred to as *reference set*, and typically consists of the following elementary components.

- *Diversification generation method* An arbitrary solution is used as a starting point (or seed) to generate a set of diverse trial solutions. There are a number of ways to implement this process such as using experimental design in statistics or taking advantage of the problem structure.

- *Improvement method* This method is concerned with solution improvement in two aspects: feasibility and quality. The improvement method generally incorporates a heuristic procedure to transform an infeasible solution into a feasible one, or to transform an existing feasible solution to a new one with a better objective value.
- *Reference set update method* A small reference set containing high quality and mutually diverse solutions is dynamically updated throughout the evolution process. Subsets of the reference set are used to produce new solutions that compete with the incumbent members of the reference set for inclusion as new members. A simple option to update the reference set is to include the best solution as the first member and then select the remaining members according to their solution quality relative to the objective value. However, the next solution to be selected must satisfy the minimum diversity criterion requesting that the minimum distance between this solution and the members currently in the reference set is greater than a specified threshold.
- *Subset generation method* Subsets from the reference set are successively generated as a basis for creating combined solutions. The simplest implementation is to generate all 2-element subsets consisting of exactly two reference solutions. Campos et al. (2001) have empirically shown that the subset generation method employing 2-element subsets can be quite effective, though systematic procedures for generating key subsets consisting of larger numbers of elements invite further investigation.
- *Solution combination method* Each subset produced by the subset generation method is used to create one or more combined solutions. The combination method for solutions represented by continuous variables employs linear combinations of subset elements, not restricted to convex combinations. The weights are systematically varied each time a combined solution is generated.

SS manifested a wealth of successful applications (Marti, 2006), ranging from resource assignment, flow shop scheduling, network routing, software testing, to bioinformatics.

3. The proposed methods

Next, we propose our methods for tackling PPR using PSO and SS, respectively.

3.1 PSO for PPR

This method is based on our previous work (Yin, 2006). To apply PSO for solving the PPR problem, we device specific features as follows.

Particle Coding Scheme

We encode the affine transformation parameters, namely, the rotation angle θ, the scale factor s, and the translation offsets t_x and t_y in the particle representation, i.e., the particle vector looks like

$$P = (\theta, s, t_x, t_y)^T \tag{10}$$

where each parameter value is a random real number and is restricted by an appropriate range. In particular, $0^\circ \le \theta \le 360^\circ$, $0 < s \le 10$, and $-200 \le t_x, t_y \le 200$ are appropriate for a large number of applications. As such each particle encoded in this way corresponds to one set of affine transformation parameters to align the point patterns. During evolution, the particles are constrained to move in the same ranges as they are initialized. When the particles reach the boundary constraints, they are set to boundary values.

Fitness Evaluation and Bounding Criterion

In PSO, the solution quality, or *fitness*, delivered by each particle is evaluated. The two alternative registration distance measures, AHD and PHD, can be used for this purpose. Since these measures are error functions, a particle delivering a smaller AHD or PHD value is considered to be superior to the other particles with larger AHD or PHD values. As such the *pbest* and *lbest* can be determined according to the fitness values of all particles.

Here we propose a bounding criterion to speedup the computation for determining *pbest* and *lbest*. The formulae (7)-(9) of particle movement refer to representations of *pbest*, *lbest*, and the particle itself, not directly to their fitness values. We propose to use this property for saving computation time. Since the fitness value of a particle is only used for updating of *pbest* and *lbest*, we can use the fitness value of the incumbent $pbest_i$ as an error upper bound to terminate the fitness computation of particle i. More precisely, the computation of AHD involves an error summation over point registration (see Eqs. (2) and (3)) and the computation of PHD is also resulted from the maximum of two sub-error measures (see Eq. (6)), both of which are a value-increasing computation. Hence, we can terminate the error computation for particle i upon the time the intermediate error value exceeds the fitness value of the incumbent $pbest_i$, and go directly to the fitness evaluation of the next particle. Also, only those $pbest_i$ that have been updated at the current iteration need to be compared to associated *lbest* for its possible updating. The use of bounding criterion can save the computational time significantly.

The Algorithm

The proposed algorithm is summarized in Fig. 1. Initially, a swarm of particles are created at random and each of which is a vector corresponding to an instance of transformation parameters to the underlying problem. Then, the particle movement is repeated until a maximal number of iterations have been passed. During each iteration, the particle individual best and swarm's best positions are determined using the bounding criterion. The particle adjusts its position based on the individual experience ($pbest_i$) and the swarm intelligence ($lbest_i$). When the algorithm is terminated, the best of all $pbest_i$ and the corresponding fitness value are output and considered as the optimal transformation parameters and the alignment error.

1. Initialize.
   1.1 Randomly generate M particles, P_1, P_2, …, P_M, according to Eq. (10).
2. Repeat until a given maximal number of iterations is achieved.
   2.1 Determine $pbest_i$ and $lbest_i$, i = 1, 2, …, M using the bounding criterion.
   2.2 Update velocities v_{ij} using Eqs. (7) and (9)
   2.3 Update particles' positions using Eq. (8).
3. Output the best of all $pbest_i$ and the corresponding fitness value as the optimal transformation parameters and the alignment error.

Fig. 1. PSO algorithm for the PPR problem.

Unless specified, in all of the experiments presented in the next section we use a swarm of 20 particles, acceleration constants $c_1 = c_2 = 2.05$, and constriction factor K is equal to 0.729. These parameter values are determined empirically and conform to most settings in existing applications.

3.2 SS for PPR

Now we describe the implementation details of SS for solving the PPR problem.

Solution Coding Scheme and Improvement

The nature parameter coding scheme (10) can also be employed in SS to represent a candidate solution x, respecting the appropriate ranges of parameters. At this stage, the SS method generates a set of random solutions and improves their quality by perturbation. For each random solution, the improvement method sequentially selects each parameter in turn and alters its value by an arbitrary small deviation. If the fitness of the random solution is improved, the altered solution replaces the random solution. Otherwise, the random solution is restored. This process is repeatedly performed until the current solution cannot be further improved by examining all parameters once. As such, we obtain a set of local optimal solutions from the initial random solutions.

The initial reference set is built by selecting elements from the local optimal solutions based on the minimum diversity criterion. The best local optimal solution is firstly included in the reference set, the selection of the next best member, however, should satisfy the minimum diversity criterion, i.e., the minimum distance between the solution to be selected and all the members currently contained in the reference set is greater than a specified threshold. Therefore, the quality and diversity of the reference set are above a critical level.

Subset Generation and Solution Combination

Inspired by previous comparative researches, we implement 2-element subset generation and linear solution combination. In other words, every subset of reference set containing exactly two reference solutions is subject to linear solution combination. Given a subset containing two reference solutions x^1 and x^2, from which three new solutions x^3, x^4 and x^5 are generated as follows.

$$x^3 = x^1 - r(x^2 - x^1) \tag{11}$$

$$x^4 = x^1 + r(x^2 - x^1) \tag{12}$$

$$x^5 = x^2 + r(x^2 - x^1) \tag{13}$$

where $r \in (0, 1)$. Hence, the generated solutions x^3, x^4 and x^5 are located on the line determined by the two reference solutions x^1 and x^2 if r is constant. Nevertheless, we adopted different values of r along various parameter dimensions to expand the search beyond the line.

Bounding Criterion and Reference Set Update

The bounding criterion used in our PSO method is also enforced here to expedite the process. For each candidate solution produced by the solution combination method, we evaluate its fitness and use the worst fitness of current members in reference set as the upper bound. That is, the fitness evaluation of the candidate solution is terminated if the intermediate fitness value exceeds the upper bound and this candidate solution is abandoned.

Assume that k_1 feasible solutions (satisfying the bounding criterion) are produced by the solution combination method and the reference set contains k_2 solutions. Our reference set update method is conducted as follows. The best $k_2/2$ solutions in the pool of new solutions and the reference set are selected first into the new reference set. For each of the rest $k_1 + k_2/2$ solutions, the minimum distance to the current members in the new reference set is computed. Then, the solution with the maximum of these minimum distances is added to

the new reference set. This max-min selection process is repeated until the new reference set contains k_2 solutions. Thus, both the quality and the diversity of the reference set members are guaranteed.

The Algorithm

Our SS algorithm proceeds as follows. The diversification generation method and improvement method are applied to create a set of random solutions that satisfy a critical level of diversity and quality. This set is used to produce the initial reference set. Every 2-element subset of the reference set is generated and used to produce three new solutions by the solution combination method. Only quality new solutions passing the bounding criterion are remained. The remained solutions are further improved by the improvement method. The reference set is then updated by comparing the new solutions to the solutions currently in the reference set according to the reference set update method. The process is repeated until the reference set cannot be further updated. The SS algorithm is summarized in Fig. 2.

1. Initialize.
   - 1.1 Create a set *P* of local optimal solutions obtained by altering a set of random solutions using the improvement method.
   - 1.2 Build the initial reference set, denoted by *RefSet*, by selecting members from *P* based on the minimum diversity criterion.
2. Repeat until *RefSet* cannot be further updated.
   - 2.1 Generate all 2-element subsets of *RefSet*.
   - 2.2 Use the members of each 2-element subset to generate three new solutions by applying Eqs. (11)-(13).
   - 2.3 Remain quality new solutions using the bounding criterion and improve them by the improvement method.
   - 2.4 Update *RefSet* by the max-min selection process.
3. Output the best member of *RefSet* and the corresponding fitness value as the optimal transformation parameters and the alignment error.

Fig. 2. SS algorithm for the PPR problem.

4. Experimental results

In this section, we present the experimental results and analyze the computational performance. The platform of the experiments is a PC with a 1.8 GHz CPU and 192 MB RAM. All programs are coded in C++ language.

4.1 Synthetic datasets

To evaluate the performance of competing algorithms, several synthetic datasets are prepared. Fig. 3(a) shows a typical point set, referred to as pattern *A*, consisting of 250 points generated at random. Four scenarios widely seen in real-world applications are used to generate testing patterns as shown in Figs. 3(b)-3(e) to match with pattern *A*.

- Scenario RST: A testing pattern is generated by applying to pattern *A* with an affine transformation consisting of rotation, scaling, and translation. In particular, the transformation parameters (θ, s, t_x, t_y) are set to (30°, 1.5, 19.0, 42.0). The resulting testing pattern, referred to as pattern *B*, is shown in Fig. 3(b). It can be formulated by

Pattern B = T[Pattern A]

- Scenario RSTP: In addition to applying Scenario RST, a random perturbation quantity is added to the coordinates of each transformed data point. The random perturbation quantity is generated uniformly within one percent of the maximum positional span along every coordinate axis dimension and the resulting testing pattern is shown in Fig. 3(c). The formulation is as follows.

Pattern B = T[Pattern A] + perturbation

- Scenario RSTPA: Besides applying Scenario RSTP, we augment the size of the resulting point pattern by 20% by adding 50 spurious random points and thus yielding a 300-point pattern (see Fig. 3(d)), *viz.*,

Pattern B = T[Pattern A] + perturbation + 20% points

- Scenario RSTDA: First, Scenario RST is applied to pattern A. Then, randomly select 20% of the points and remove them. Generate the same number of spurious points and add them to the point pattern (see Fig. 3(e)). The formulation is as follows.

Pattern B = T[Pattern A] - 20% points + 20% points

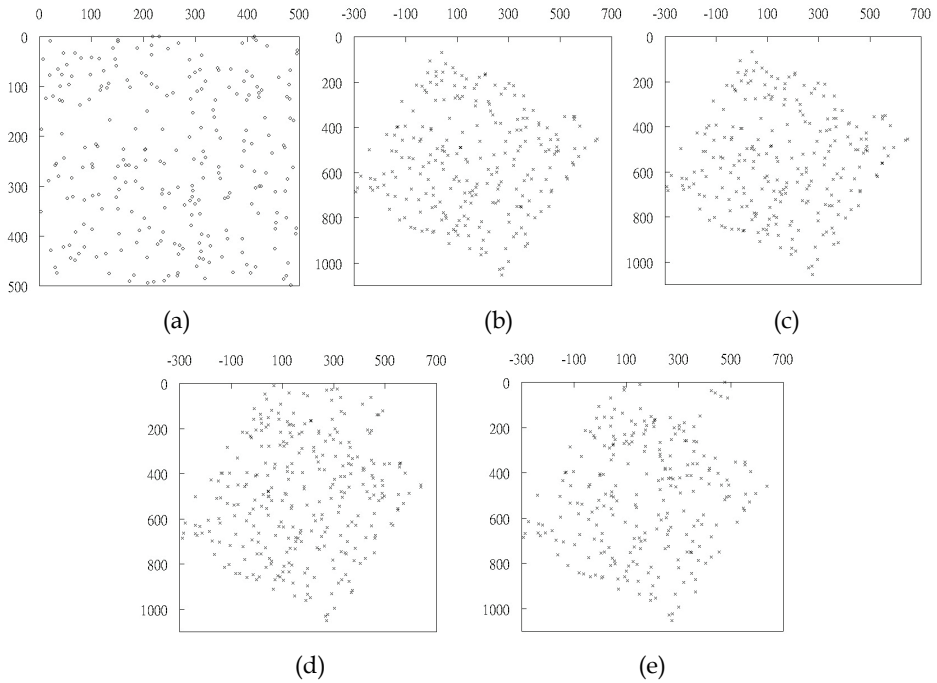

Fig. 3 Synthetic point patterns. (a) Pattern A, (b) a testing pattern generated by Scenario RST, (c) a testing pattern generated by Scenario RSTP, (d) a testing pattern generated by Scenario RSTPA, and (e) a testing pattern generated by Scenario RSTDA.

Following the protocol generating the previous dataset, we can create other datasets with different numbers of points (n), in particular, point patterns having 50, 250, and 500 points under the previously noted real scenarios are built.

4.2 Empirical study

The comparative performance of the proposed evolutionary algorithms is analyzed by comparing to competing algorithms under various testing scenarios. In particular, we have implemented two traditional evolutionary algorithms, namely, genetic algorithm (GA) and simulated annealing (SA). GA is a population-based evolutionary algorithm which explores the search space using a number of individual agents, called *chromosomes*. We implemented the GA with the same coding scheme, initialization ranges, and fitness evaluation as used to implement our proposed algorithms. In addition to the broadly used genetic operations, namely the selection, crossover, and mutation, we further emploied fitness scaling and elitist strategy (Goldberg, 1989) to enhance the performance of the GA. On the other hand, SA is an evolutionary algorithm based on perturbation of the current configuration (candidate solution) in order to reach an equilibrium state, simulating the thermal annealing process. The implemented SA also uses the same coding scheme, initialization ranges, and fitness function as used by our algorithms.

The comparative performance of competing algorithms is evaluated with different datasets containing various numbers of data points (n), in particular, n = 50, 250, and 500, respectively. For a fair comparison, all competing algorithms are terminated when they have consumed 4000 times of fitness evaluations because solution fitness is the most informative element and is also the most time consuming component. In all experiments, PSO is executed with 20 particles, SS maintains a reference set containing 20 elite solutions, and GA is conducted with 20 chromosomes. It is worth noting that SA is a single agent search algorithm instead of a population-based one, we thus let SA execute with 4000 iterations. The numerical results for each dataset are the mean value and the standard deviation (σ) from 30 independent runs and they are summarized in Tables 1 and 2 (the CPU times are evaluated in seconds) for AHD and PHD measures, respectively. We have the following observations.

- Overall, PSO and SS have the best performance among all the competing algorithms in terms of both dissimilarity measures (AHD and PHD) and computational time, SA is ranked at the middle place, and GA seems to be the worst of all.
- PSO and SS are also more stable than GA and SA, by producing consistent results with smaller standard deviation values.
- For all RST testing datasets, PSO does not yield any registration errors for both AHD and PHD measures because RST incurs a complete registration with no perturbation, SS produces negligible errors, while both GA and SA entail significant amount of errors and fail to find the optimal transformation.
- For all RSTDA testing datasets, PSO produces no registration errors for the PHD measure, which means the PSO algorithm is able to find the groundtruth transformation for incomplete registration without perturbation (PSO does not generate zero error with AHD measure because AHD penalizes an incomplete registration, however, the point correspondences are still correctly identified).
- At average, PSO consumes about 80% of the CPU time required by GA and about 91% of the CPU time required by SA. The average CPU time cost by SS is comparable to that

cost by PSO, but the large value of standard deviation indicates that the individual CPU time spent by SS in each independent run varies a lot. This is because the CPU time consumed by SS depends on the number of times the reference set is updated and this number is determined by the time upon which good quality solutions are generated, which varies with different runs.

		PSO		SS		GA		SA	
n	Scenario	AHD (σ)	Time (σ)	AHD (σ)	Time (σ)	AHD (σ)	Time (σ)	AHD (σ)	Time (σ)
50	RST	0.0 (0.0)	8.8 (0.1)	0.1 (0.1)	8.8 (0.6)	9.2 (0.5)	11.5 (0.1)	0.3 (0.3)	9.9 (0.3)
	RSTP	11.2 (0.0)	8.6 (0.1)	11.2 (0.0)	7.6 (2.6)	14.3 (1.1)	10.8 (0.1)	11.6 (0.5)	9.7 (0.2)
	RSTPA	14.1 (0.0)	10.3 (0.1)	14.2 (0.0)	6.6 (1.6)	14.9 (1.2)	12.9 (0.2)	14.6 (0.3)	11.2 (0.1)
	RSTDA	204.2 (0.0)	8.7 (0.1)	204.4 (0.1)	6.2 (2.3)	211.4 (6.2)	10.9 (0.1)	205.9 (1.9)	9.5 (0.1)
250	RST	0.0 (0.0)	206.4 (3.0)	0.0 (0.0)	155.6 (2.6)	9.5 (0.7)	262.1 (2.5)	0.9 (1.3)	237.0 (2.6)
	RSTP	10.8 (0.0)	206.4 (1.6)	10.8 (0.1)	153.7 (1.3)	40.9 (2.4)	262.8 (1.6)	11.4 (0.5)	229.7 (1.9)
	RSTPA	13.0 (0.0)	245.4 (0.2)	13.5 (0.3)	189.7 (3.6)	29.5 (1.7)	309.4 (1.8)	16.1 (2.3)	271.1 (2.5)
	RSTDA	112.3 (0.0)	204.9 (0.2)	112.5 (0.2)	155.7 (4.3)	123.9 (3.6)	263.9 (2.4)	113.4 (1.1)	224.6 (1.5)
500	RST	0.0 (0.0)	821.2 (2.1)	0.1 (0.1)	612.0 (7.9)	15.1 (1.3)	1060.3 (8.6)	0.8 (0.6)	895.7 (1.9)
	RSTP	11.4 (0.0)	817.7 (1.8)	11.5 (0.1)	597.0 (11.2)	15.9 (1.1)	1059.0 (9.3)	13.1 (2.3)	895.7 (3.6)
	RSTPA	55.3 (0.0)	979.3 (2.4)	55.3 (0.0)	692.0 (6.1)	72.6 (3.4)	1249.5 (9.2)	60.7 (3.9)	1077.6 (13.1)
	RSTDA	53.9 (0.0)	816.6 (2.0)	53.9 (0.1)	570.8 (10.5)	62.8 (3.0)	1042.9 (10.4)	55.9 (0.8)	893.9 (2.3)

Table 1. Comparative performances of the PSO, SS, GA and SA algorithms with respect to the AHD measure and the used computational time (in seconds).

		PSO		SS		GA		SA	
n	Scenario	PHD (σ)	Time (σ)	PHD (σ)	Time (σ)	PHD (σ)	Time (σ)	PHD (σ)	Time (σ)
50	RST	0.0 (0.0)	12.1 (0.2)	0.2 (0.1)	11.7 (2.6)	20.4 (1.4)	14.1 (0.1)	2.0 (2.4)	13.0 (0.1)
	RSTP	16.6 (0.1)	11.8 (0.1)	17.4 (0.1)	14.1 (5.7)	24.5 (1.6)	14.3 (0.2)	20.3 (2.0)	12.9 (0.1)
	RSTPA	22.4 (0.6)	14.1 (0.1)	23.8 (0.6)	16.0 (2.1)	36.4 (2.2)	17.2 (0.3)	24.8 (1.8)	15.4 (0.1)
	RSTDA	0.0 (0.0)	11.8 (0.1)	0.1 (0.1)	12.3 (5.3)	29.5 (1.9)	16.9 (0.2)	1.4 (1.4)	12.9 (0.1)
250	RST	0.0 (0.0)	288.3 (1.8)	0.3 (0.2)	312.0 (5.3)	149.7 (4.8)	374.0 (2.1)	1.0 (1.1)	315.3 (1.3)
	RSTP	19.8 (0.1)	289.3 (1.7)	20.5 (0.2)	306.7 (2.2)	37.2 (2.1)	347.9 (2.3)	22.0 (1.0)	314.0 (0.3)
	RSTPA	24.1 (1.5)	341.6 (1.9)	25.8 (0.4)	369.6 (14.9)	168.9 (6.1)	427.9 (2.9)	30.4 (5.0)	376.8 (1.8)
	RSTDA	0.0 (0.0)	283.8 (0.2)	0.1 (0.0)	269.8 (7.6)	116.8 (3.9)	346.0 (1.9)	2.4 (2.6)	312.8 (1.4)
500	RST	0.0 (0.0)	1148.8 (2.2)	0.1 (0.1)	1132.3 (26.3)	9.7 (0.7)	1386.7 (6.2)	1.2 (1.0)	1259.0 (1.1)
	RSTP	18.4 (0.1)	1147.3 (2.7)	19.0 (0.1)	1215.4 (21.7)	24.5 (1.2)	1382.2 (8.1)	22.4 (2.4)	1259.3 (3.1)
	RSTPA	53.7 (2.1)	1385.5 (8.0)	56.6 (0.8)	1305.2 (35.8)	70.4 (4.5)	1656.1 (9.3)	71.7 (14.7)	1497.4 (2.2)
	RSTDA	0.0 (0.0)	1130.9 (2.7)	0.1 (0.1)	1090.8 (13.6)	76.2 (3.8)	1366.8 (8.6)	0.9 (0.9)	1234.7 (2.2)

Table 2. Comparative performances of the PSO, SS, GA and SA algorithms with respect to the PHD measure and the used computational time (in seconds).

Fig. 4 shows the registration results between pattern *A* and pattern *B* with all testing scenarios obtained using the proposed PSO method. The registration results obtained using the proposed SS method is very similar to that obtained by PSO, so we omit the illustration

of SS for saving space. For each testing scenario, PSO is performed with AHD and PHD dissimilarity metrics, respectively. For the Scenario RST (see Figs. 4(a) and 4(b)) and the Scenario RSTP (see Figs. 4(c) and 4(d)), the PSO can find the complete registration between the two patterns and the one-to-one correspondence relationship is correctly identified. For the Scenario RSTPA (see Figs. 4(e) and 4(f)) and the Scenario RSTDA (see Figs. 4(g) and 4(h)), the PSO derives the incomplete registration. The point patterns are appropriately aligned for both scenarios and the groundtruth point registration correspondences are found. Note that in all testing scenarios both AHD and PHD measures work well with the proposed method.

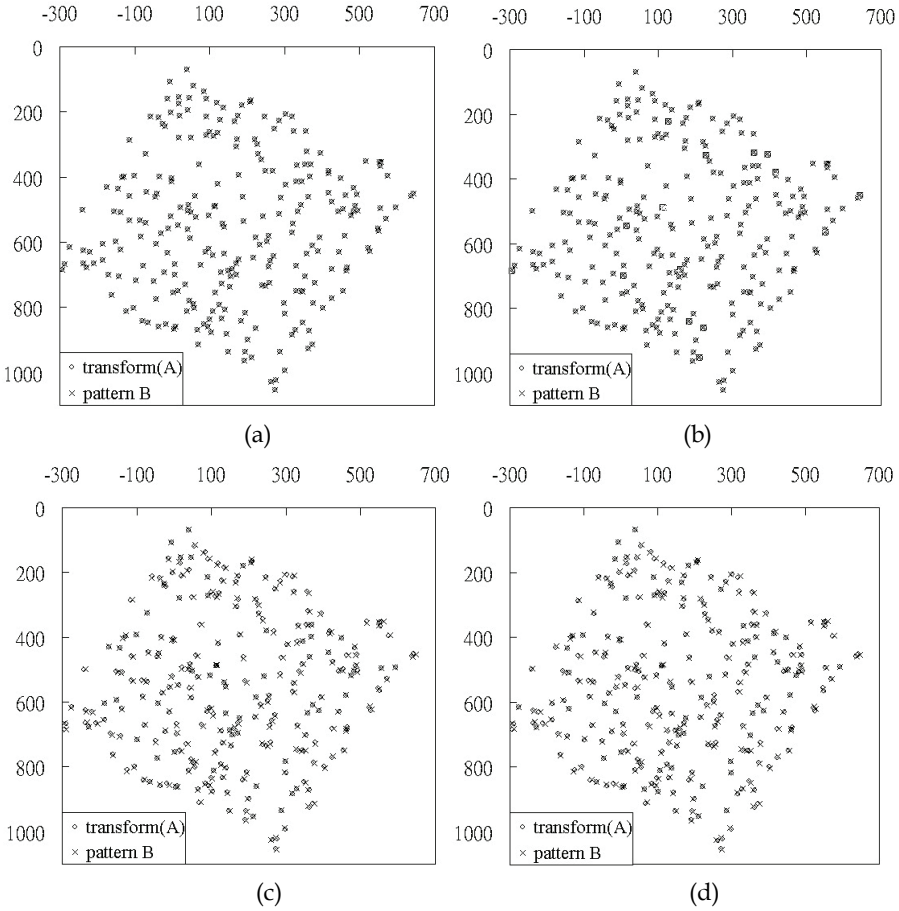

Fig. 4 Registration results obtained using the PSO method for all testing scenarios. (a) Scenario RST with AHD metric, (b) Scenario RST with PHD metric, (c) Scenario RSTP with AHD metric, (d) Scenario RSTP with PHD metric, (e) Scenario RSTPA with AHD metric, (f) Scenario RSTPA with PHD metric, (g) Scenario RSTDA with AHD metric, and (h) Scenario RSTDA with PHD metric.

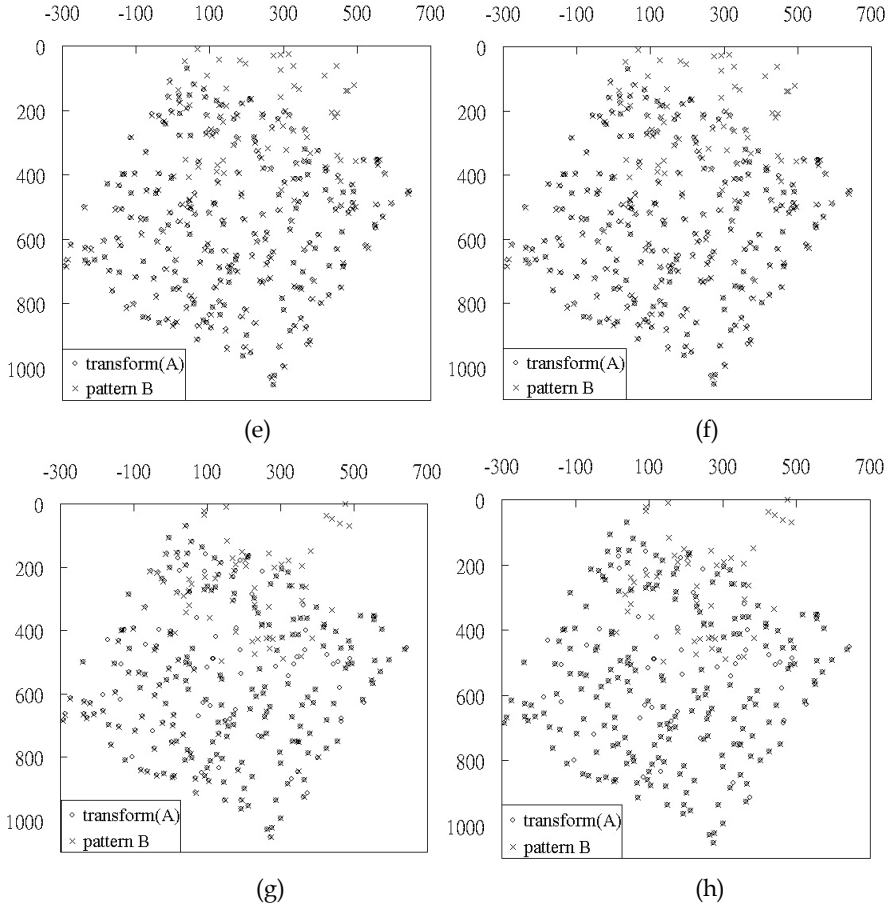

Fig. 4 Registration results obtained using the PSO method for all testing scenarios (continued.)

5. Conclusion

This chapter investigates the strengths and weaknesses of PPR approaches based on modern evolutionary algorithms, in particular, the particle swarm optimization (PSO) and scatter search (SS). The experimental results manifest that PSO and SS are malleable under varying scenarios such as positional perturbations, contaminations and drop-outs from the point patterns. PSO and SS are also more effective and efficient than the methods based on genetic algorithm (GA) and simulated annealing (SA) in minimizing the registration error. The advantage of our algorithms is due to the natural metaphor, stochastic move, adaptivity, and positive feedback. Our observations disclose the truism that modern evolutionary algorithms have competitive features that provide a chance to create a solution method which is both effective and efficient and is significantly different from that created by tradition evolutionary algorithms.

6. References

Agrawal, A.; Ansari, N. & Hou, E. (1994). Evolutionary programming for fast and robust point pattern matching, *Proceedings of IEEE International Conference on Neural Networks*, pp. 1777-1782.

Ansari, N.; Chen, M.H. & Hou, E. (1990). Point pattern matching by a genetic algorithm, *Proceedings of IEEE International Conference on Industrial Electronics, II*, pp. 1233-1238.

Ansari, N.; Hou, E. & Agrawal, A. (1993). Point pattern matching by simulated annealing, *Proceedings of IEEE Regional Conference on Control Systems*, pp. 215-218.

Campos, V.; Glover, F.; Laguna, M. & Martí, R. (2001). An experimental evaluation of a scatter search for the linear ordering problem, *Journal of Global Optimization*, Vol. 21, pp. 397-414.

Carcassoni, M. & Hancock, E.R. (2003). Spectral correspondence for point pattern matching, *Pattern Recognition*, Vol. 36, pp. 193-204.

Chang, S.H.; Cheng, F.H.; Hsu, W.H. & Wu, G.Z. (1997). Fast algorithm for point pattern matching: invariant to translations, rotations and scale changes, *Pattern Recognition*, Vol. 30, pp. 311-320.

Clerc, M. & Kennedy, J. (2002). The particle swarm explosion, stability, and convergence in a multidimensional complex space, *IEEE Transaction on Evolutionary Computation*, Vol. 6, pp. 58-73.

Eberhart, R.C. & Shi, Y. (1998). Evolving artificial neural networks, *Proceedings of International Conference on Neural Networks and Brain,* pp. 5-13.

Glover, F. (1977). Heuristics for integer programming using surrogate constraints, *Decision Sciences*, Vol. 8, pp. 156-166.

Glover, F. (1998). A template for scatter search and path relinking, *Artificial Evolution*, Lecture Notes in Computer Science 1363, pp. 13-54.

Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, MA.

Goshtasby, A. & Stockman, G.C. (1985). Point pattern matching using convex hull edges, *IEEE Transaction on Systems, Man and Cybernetics*, Vol. 15, pp. 631-637.

Griffin, P.M. & Alexopoulos, C. (1991). Point pattern matching using centroid bounding, *IEEE Transaction on Systems, Man and Cybernetics*, Vol. 19, pp. 1274-1276.

Huttenlocher, D.P.; Klanderman, G.A. & Rucklidge, W.J. (1993). Comparing images using the Hausdorff distance, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 15, pp. 850-863.

Kennedy, J. & Eberhart, R.C. (1995). Particle swarm optimization, *Proceedings of IEEE International Conference on Neural Networks, IV*, pp. 1942-1948.

Marti, R. (2006). Scatter search – wellsprings and challenges, *European Journal of Operational Research*, Vol. 169, pp. 351-358.

Mount, D.M.; Netanyahu, N.S. & Moigne, J.L. (1999). Efficient algorithms for robust feature matching, *Pattern Recognition*, Vol. 32, pp. 17-38.

Ogawa, H. (1984). Labeled point pattern matching by fuzzy relaxation, *Pattern Recognition*, Vol. 17, pp. 569-573.

Olson, C.F. & Huttenlocher, D.P. (1997). Automatic target recognition by matching oriented edge pixels, *IEEE Transactions on Image Processing*, Vol. 6, pp. 103-113.

Ranade, S. & Rosenfeld, A. (1980). Point pattern matching by relaxation, *Pattern Recognition*, Vol. 12, pp. 269-275.

Shigenori, N.; Takamu, G.; Toshiku, Y. & Yoshikazu, F. (2003). A hybrid particle swarm optimization for distribution state estimation, *IEEE Transaction on Power Systems*, Vol. 18, pp. 60-68.

Starink, J.P. & Backer, E. (1995). Finding point correspondences using simulated annealing, *Pattern Recognition*, Vol. 28, pp. 231-240.

Tandon, V. (2000). Closing the gap between CAD/CAM and optimized CNC end milling, Master thesis, Purdue School of Engineering and Technology, Indiana University Purdue University Indianapolis.

Ton, J. & Jain, A.K. (1989). Registering Landsat images by point matching, *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 27, pp. 642-651.

Trelea, I.C. (2003). The particle swarm optimization algorithm: convergence analysis and parameter selection, *Information Processing Letters*, Vol. 85, pp. 317-325.

Umeyama, S. (1991). Least-squares estimation of transformation parameters between two point patterns, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13, pp. 376-380.

Wang, W.H. & Chen, Y.C. (1997). Point pattern matching by line segments and labels, *Electronic Letters*, Vol. 33, pp. 478-479.

Yin, P.Y. (2004). A discrete particle swarm algorithm for optimal polygonal approximation of digital curves, *Journal of Visual Communication and Image Representation*, Vol. 15, pp. 241-260.

Yin, P. Y. (2006). Particle swarm optimization for point pattern matching, *Journal of Visual Communication and Image Representation*, Vol. 17, pp. 143-162.

Yuen, P.C. (1993). Dominant point matching algorithm, *Electronic Letters*, Vol. 29, pp. 2023-2024.

Zhang, L.; Xu, W. & Chang, C. (2003). Genetic algorithm for affine point pattern matching, *Pattern Recognition Letters*, Vol. 24, pp. 9-19.

Pattern Recognition Techniques, Technology and Applications

Edited by Peng-Yeng Yin

ISBN 978-953-7619-24-4
Hard cover, 626 pages
Publisher InTech
Published online 01, November, 2008
Published in print edition November, 2008

A wealth of advanced pattern recognition algorithms are emerging from the interdiscipline between technologies of effective visual features and the human-brain cognition process. Effective visual features are made possible through the rapid developments in appropriate sensor equipments, novel filter designs, and viable information processing architectures. While the understanding of human-brain cognition process broadens the way in which the computer can perform pattern recognition tasks. The present book is intended to collect representative researches around the globe focusing on low-level vision, filter design, features and image descriptors, data mining and analysis, and biologically inspired algorithms. The 27 chapters coved in this book disclose recent advances and new ideas in promoting the techniques, technology and applications of pattern recognition.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821