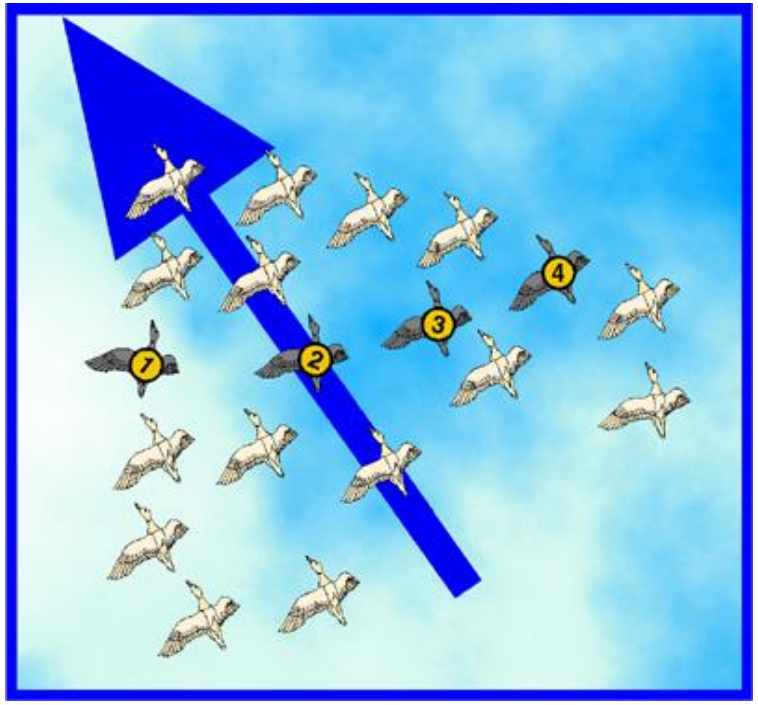
The Particle Swarm Optimization Algorithm



Summary

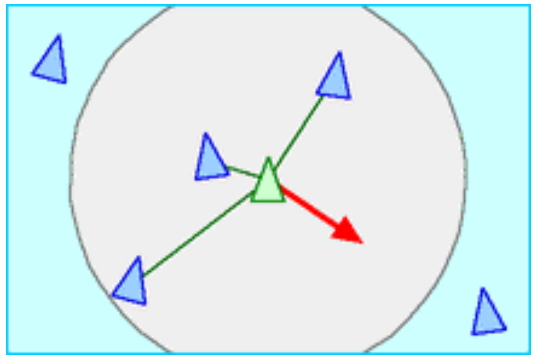
- Introduction to Particle Swarm Optimization (PSO)
  - Origins
  - Concept
  - PSO Algorithm
  - Example
  - Implementation

Introduction to the PSO: **Origins**

- Inspired from the nature social behavior and dynamic movements with communications of insects, birds and fish

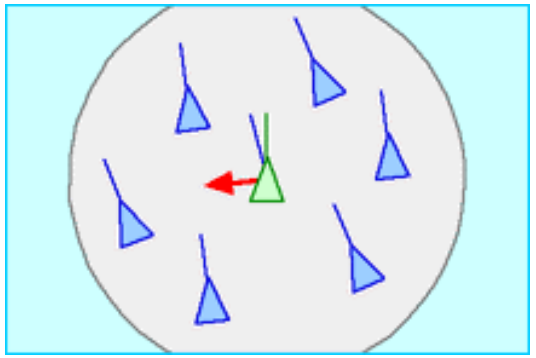
Introduction to the PSO: **Origins**

- In 1986, Craig Reynolds described this process in 3 simple behaviors:



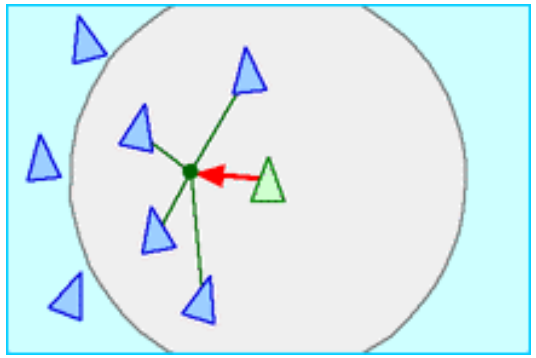
Separation

avoid crowding local flockmates



Alignment

move towards the average heading of local flockmates



Cohesion

move toward the average position of local flockmates

Introduction to the PSO: **Origins**

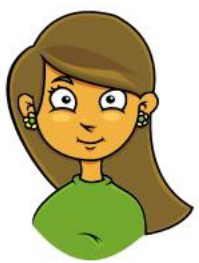
- Application to optimization: Particle Swarm Optimization
- Proposed by James Kennedy & Russell Eberhart (1995)
- Combines self-experiences with social experiences

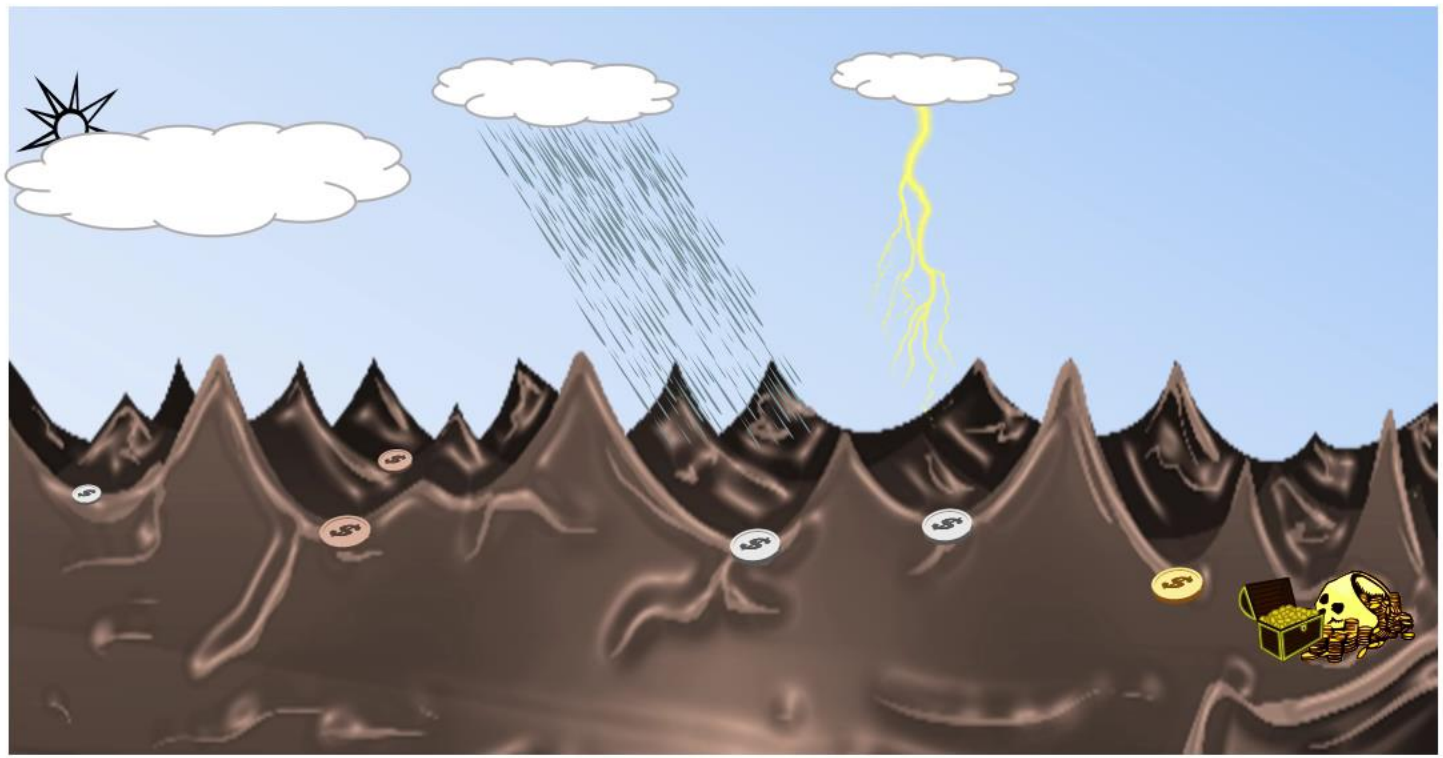
PSO search strategy

Bob

Anthony

Jennifer



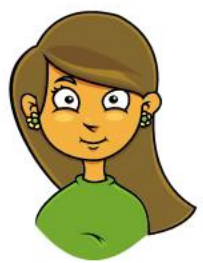


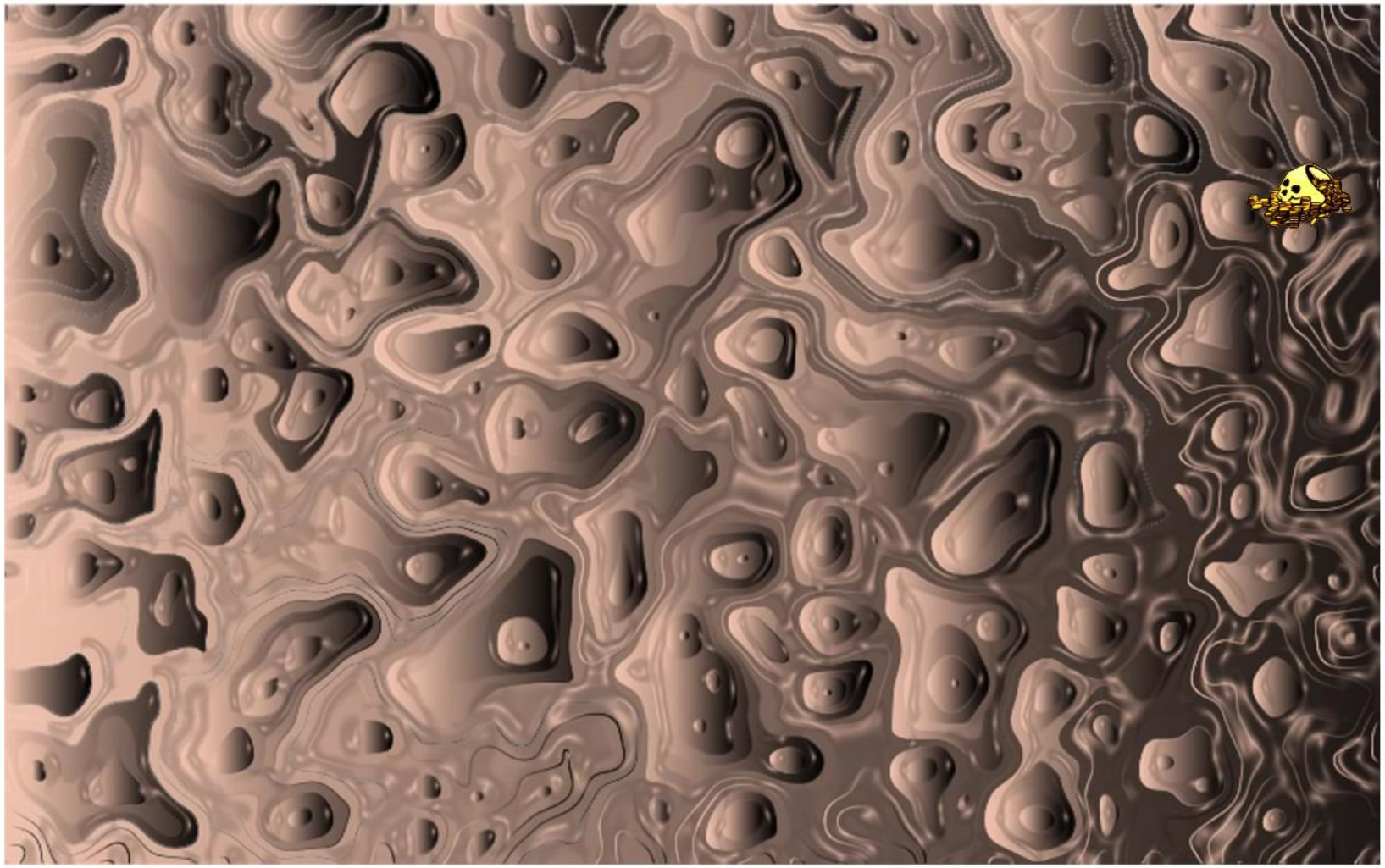
PSO search strategy

Bob

Anthony

Jennifer





PSO search strategy

Bob

Anthony

Jennifer

Personal best location

Team best location

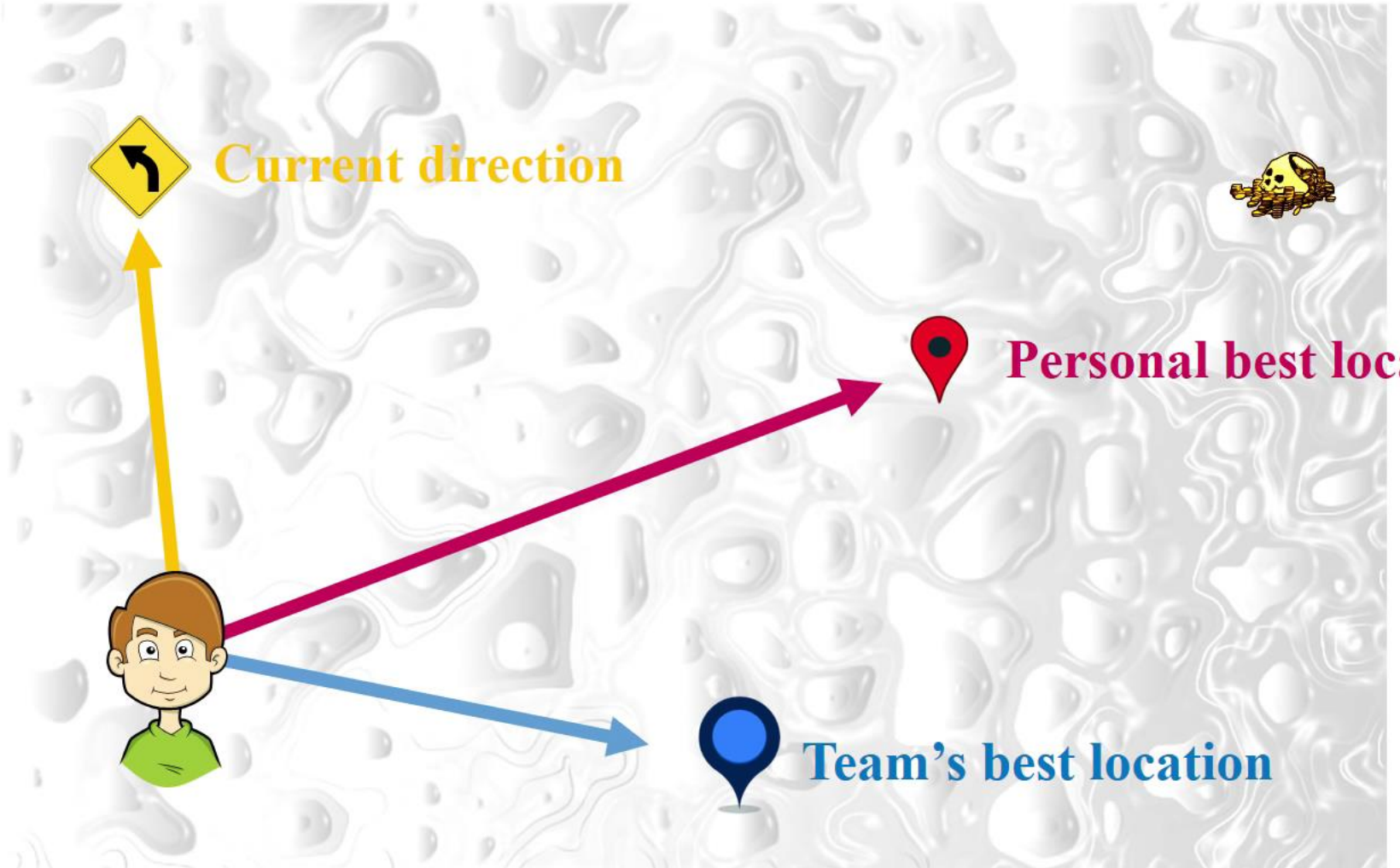
Current direction

10 km

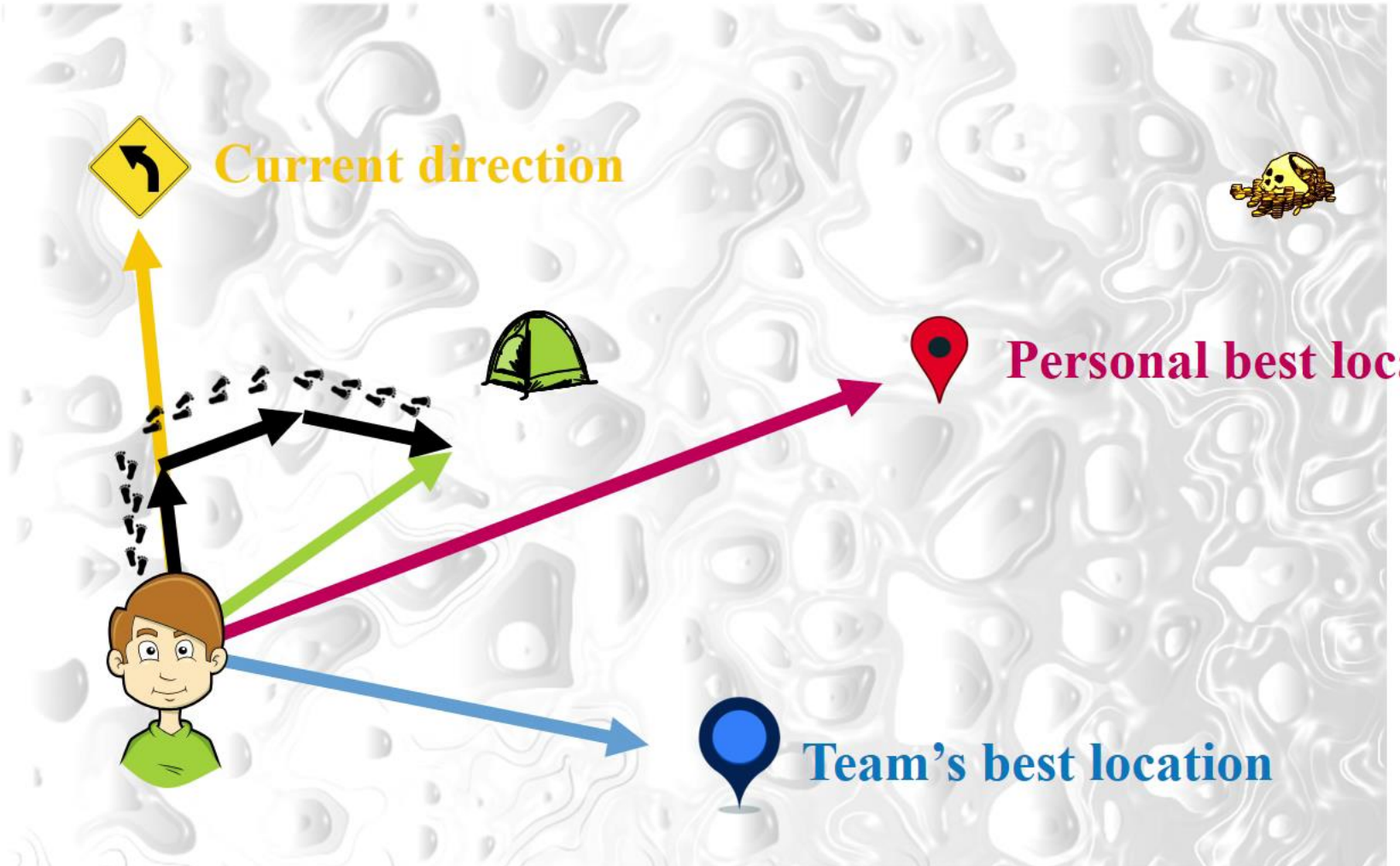
10 km

10 km

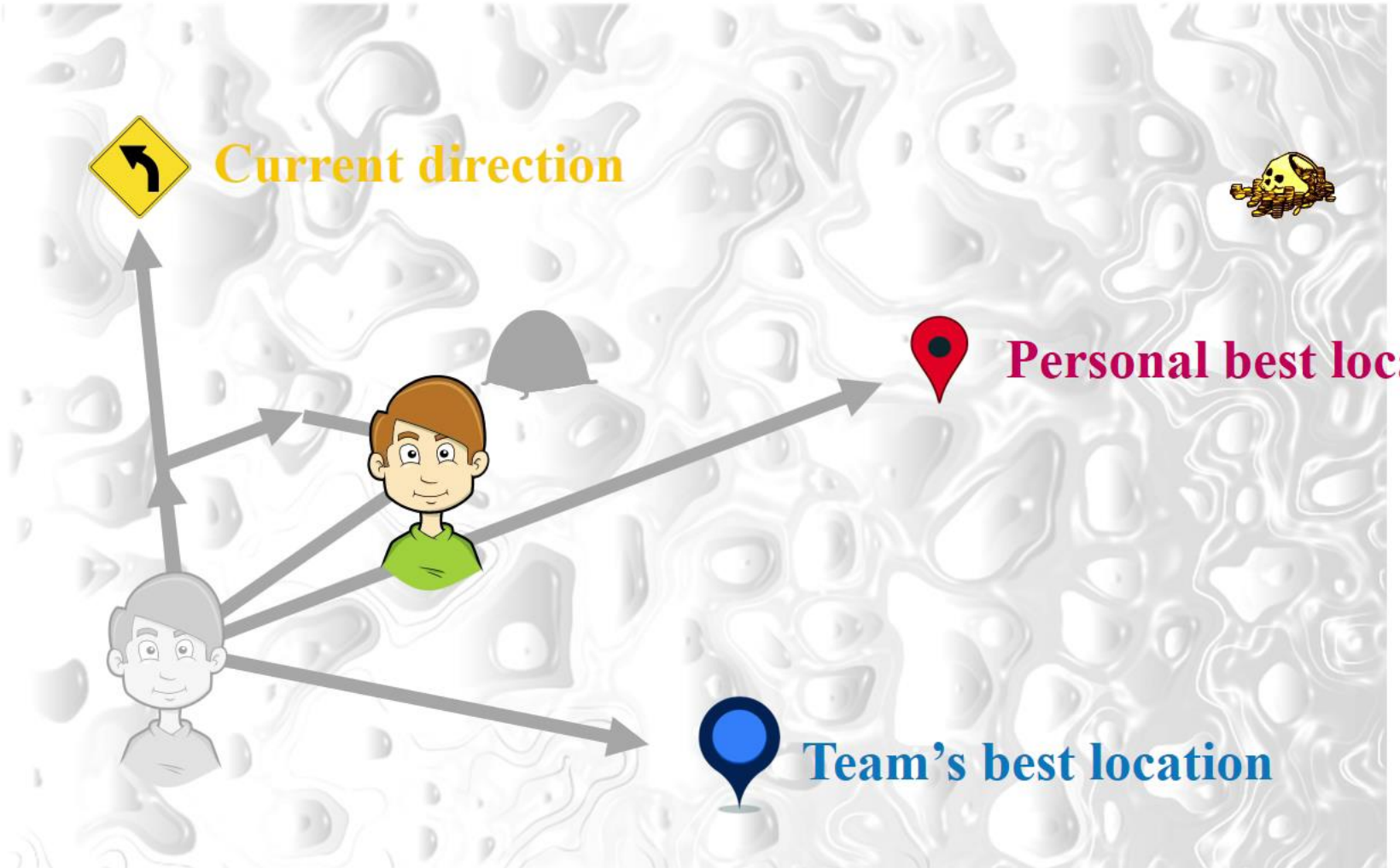
PSO search strategy



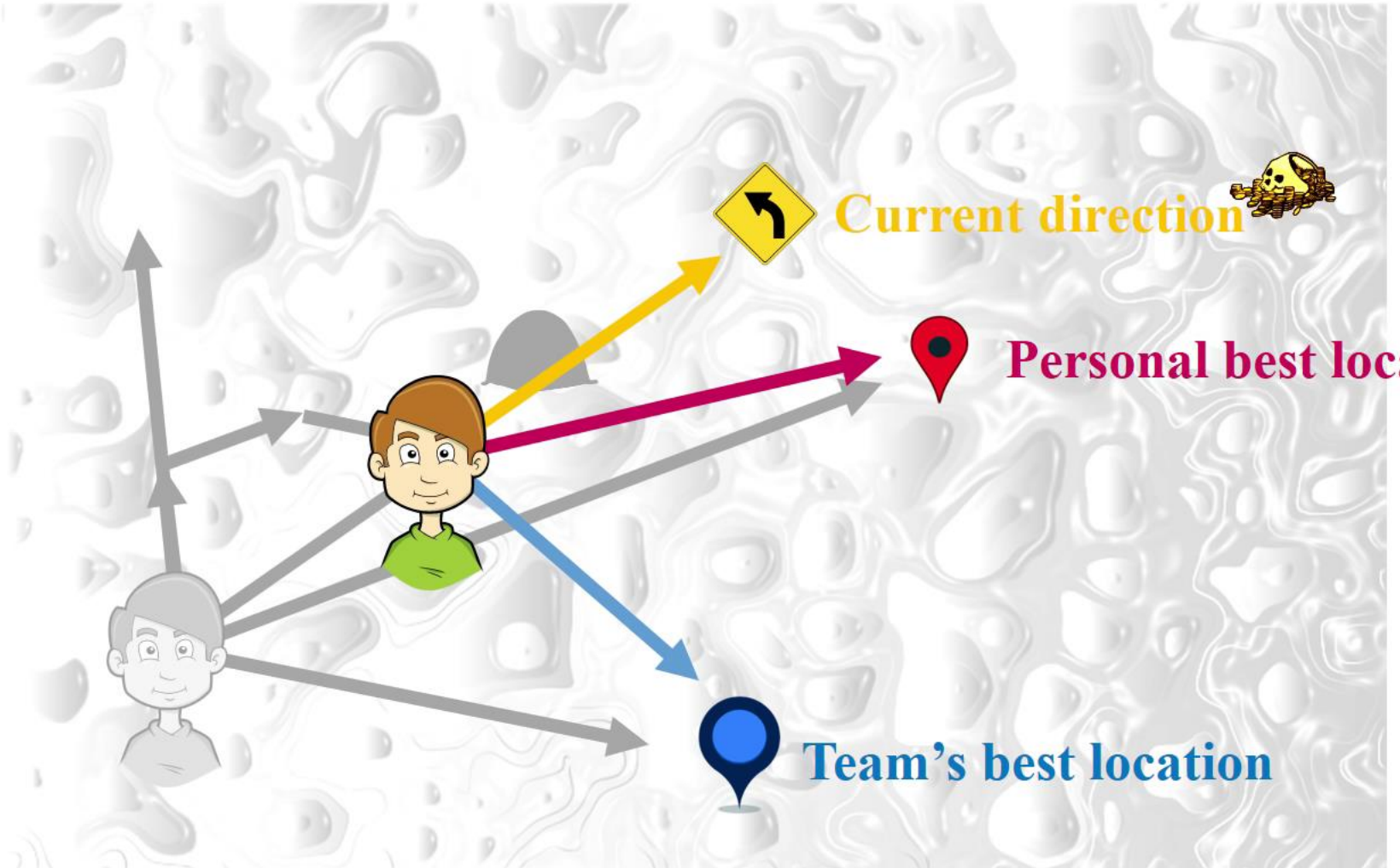
PSO search strategy



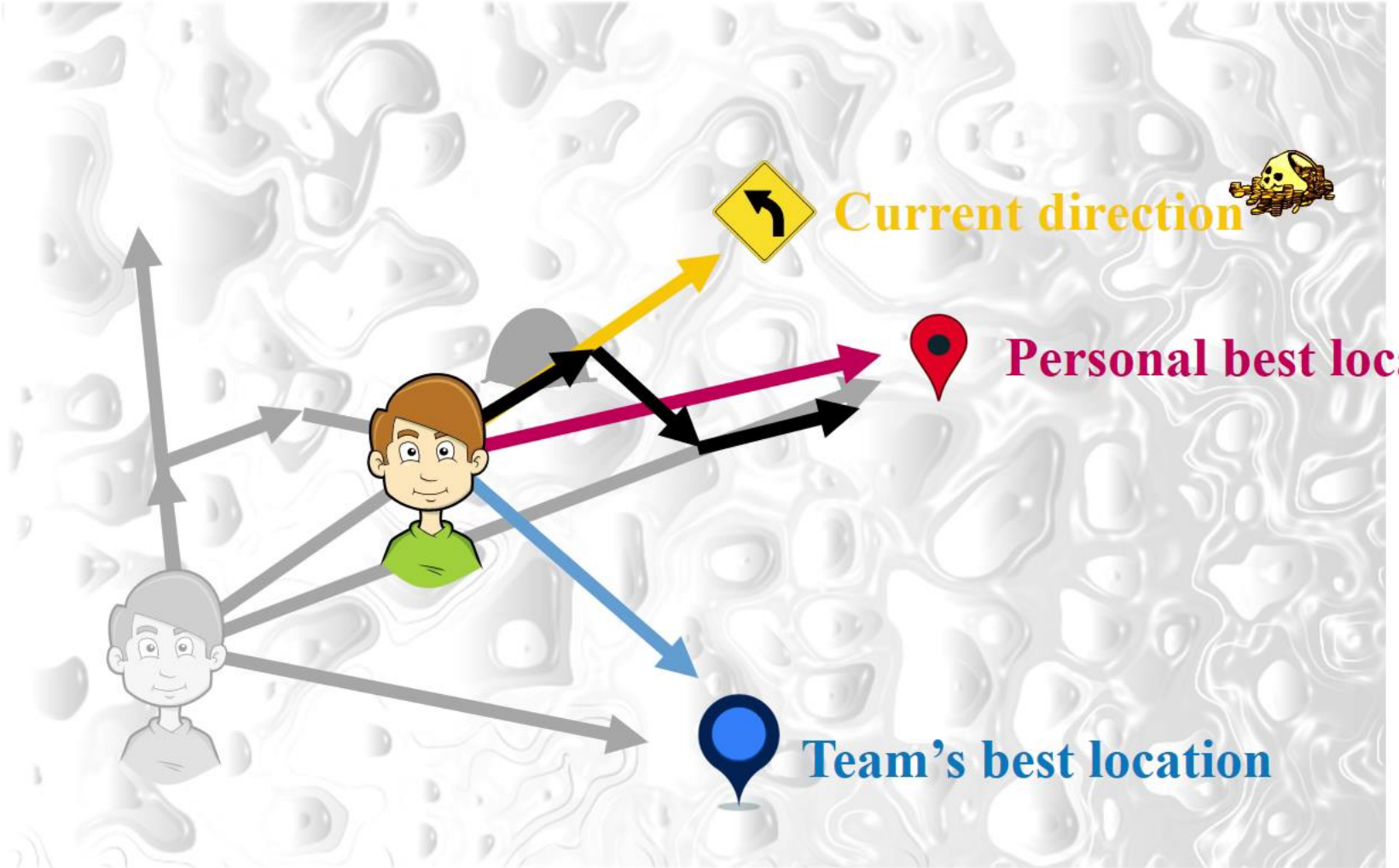
PSO search strategy



PSO search strategy



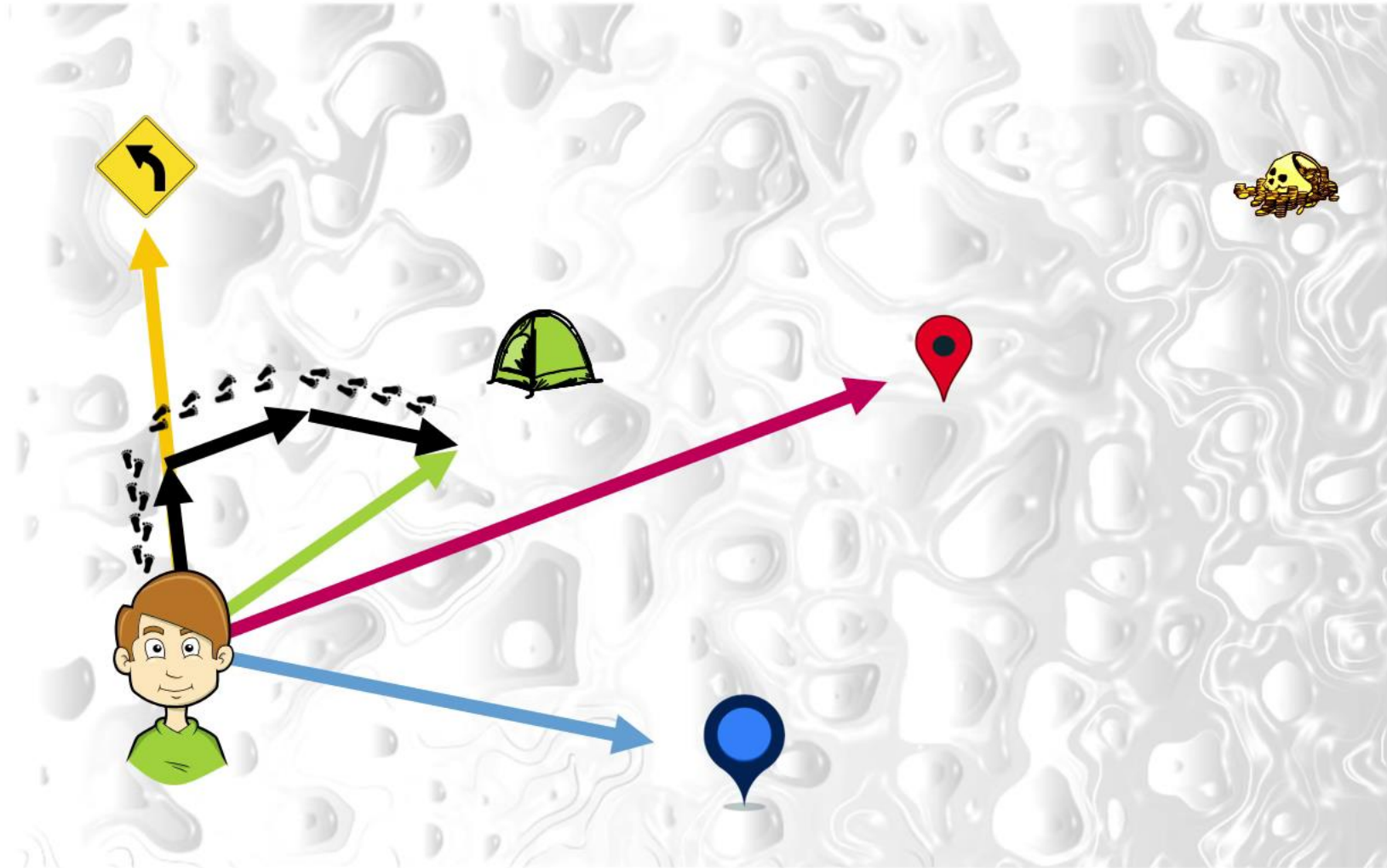
PSO search strategy



PSO search strategy

PSO search strategy

PSO search strategy



$2 \times r \times 10\ km$

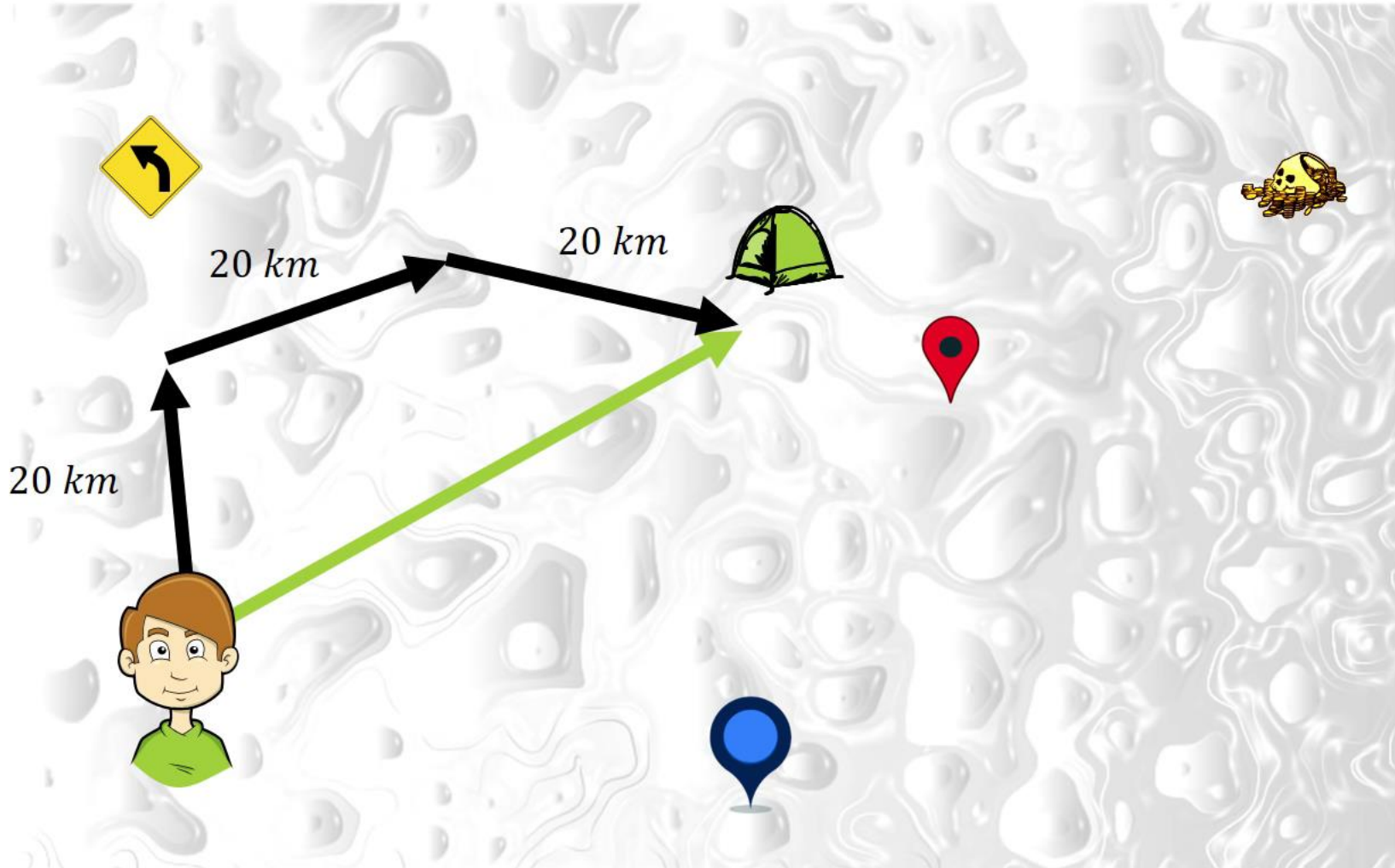
$r\ in\ [0,1]$

PSO search strategy

$2 \times r \times 10\ km$

$r\ in\ [0,1]$

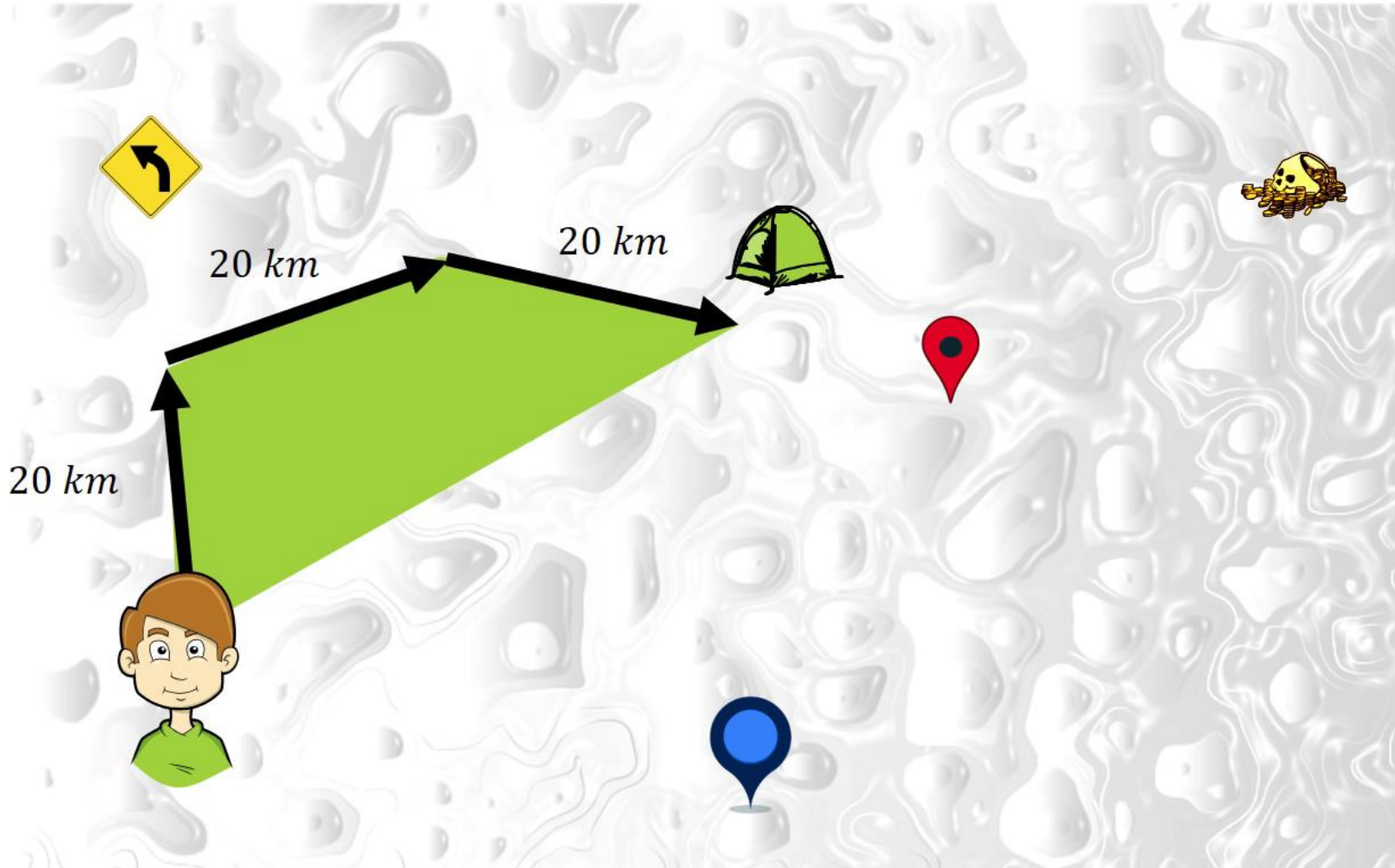
PSO search strategy



$2 \times r \times 10\ km$

$r\ in\ [0,1]$

PSO search strategy

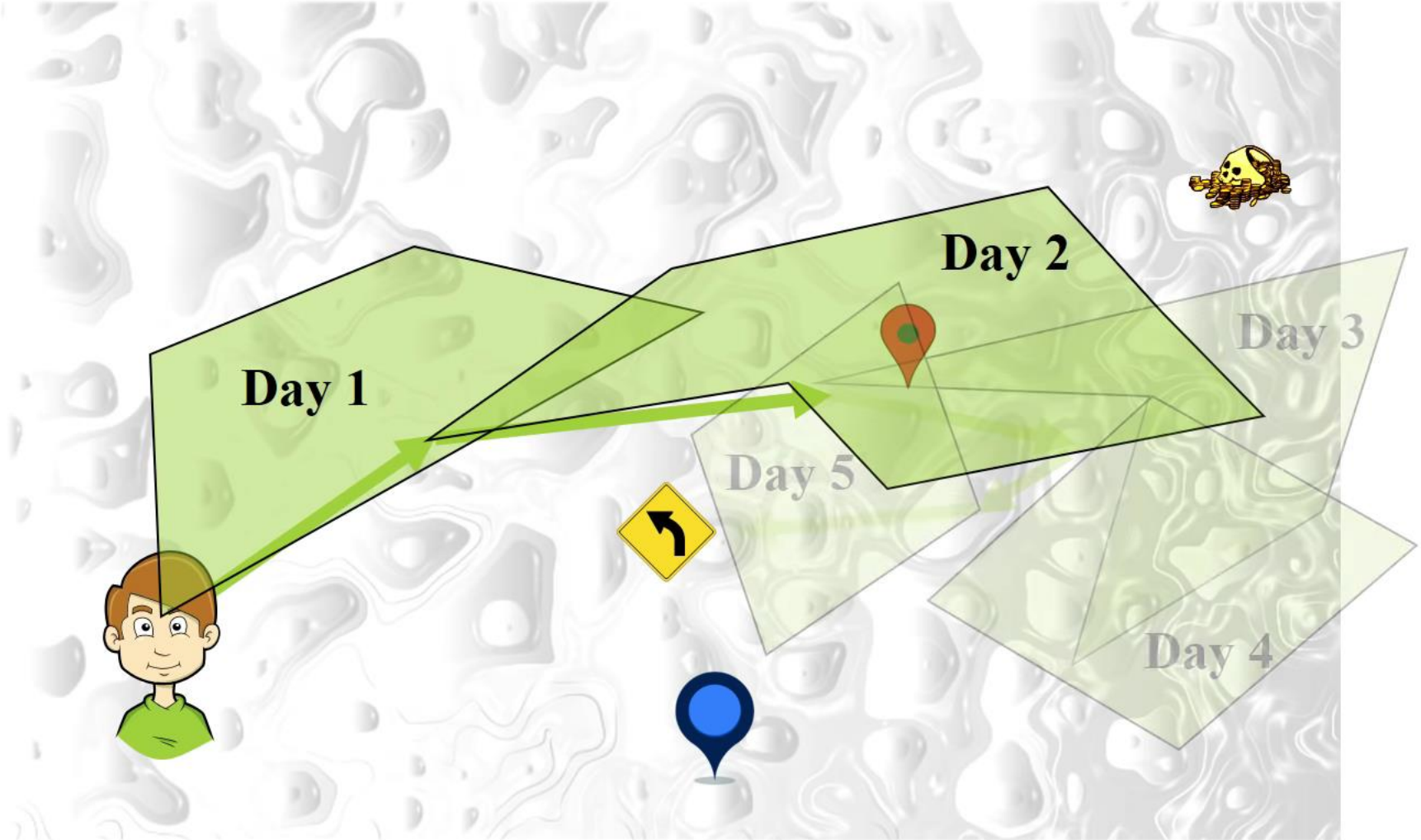


$2 \times r \times 10\ km$

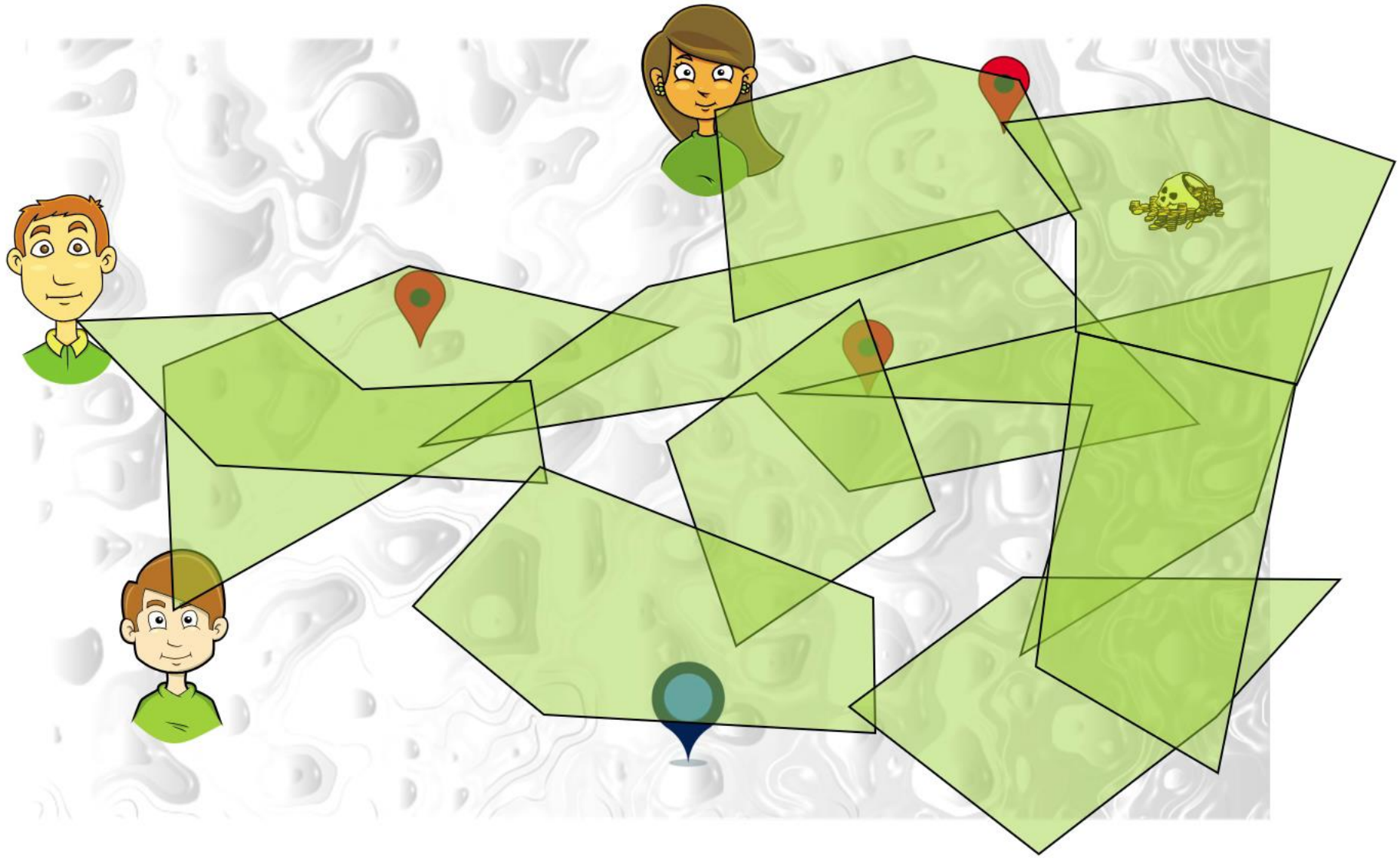
$r\ in\ [0,1]$

PSO search strategy

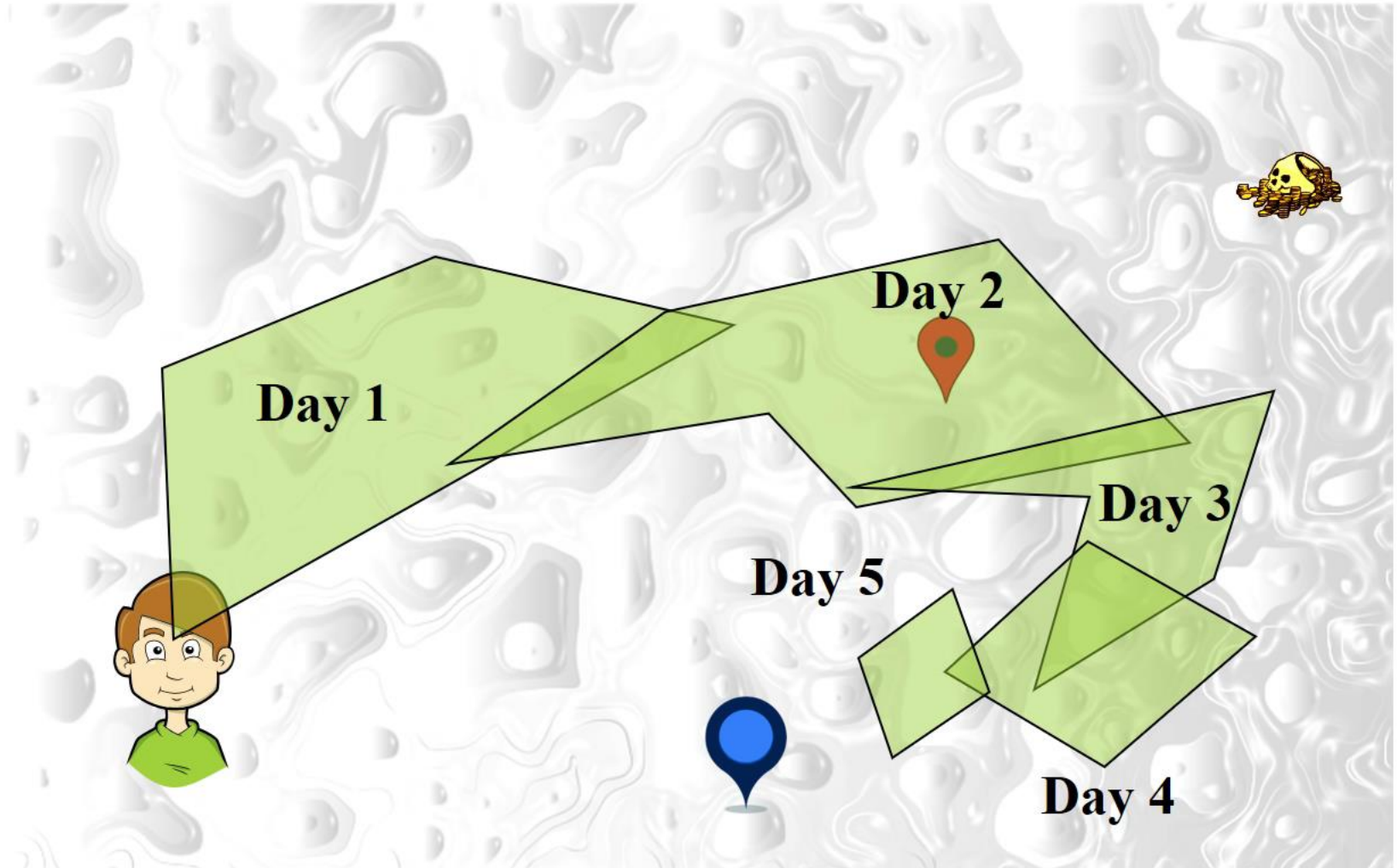
PSO search strategy



PSO search strategy



PSO search strategy



PSO search strategy

Bob

$$\overrightarrow{X_1^d} = [x_1^d, y_1^d]$$

Anthony

$$\overrightarrow{X_2^d} = [x_2^d, y_2^d]$$

Jennifer

$$\overrightarrow{X_3^d} = [x_3^d, y_3^d]$$

$$\overrightarrow{X_i^d} = [x_i^d, y_i^d, z_i^d, \dots]$$

PSO search strategy

$$\overrightarrow{V_i^{d+1}} = 2r_1 \overrightarrow{V_i^d} + 2r_2 \left(\overrightarrow{P_i^d} - \overrightarrow{X_i^d} \right) + 2r_3 \left(\overrightarrow{G^d} - \overrightarrow{X_i^d} \right)$$

Next velocity (tomorrow)

Current velocity (today)

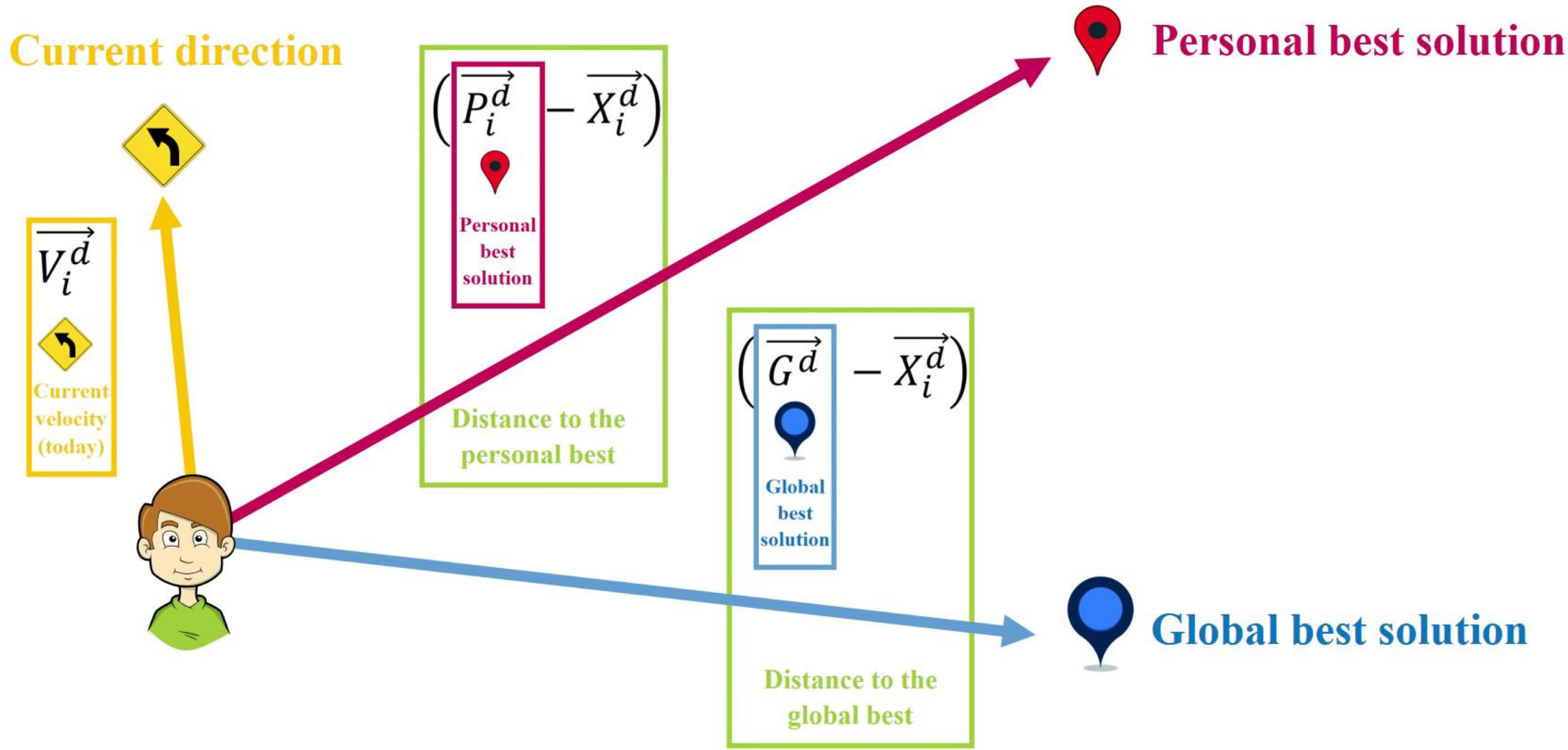
Personal best solution

Distance to the personal best

Global best solution

Distance to the global best

PSO search strategy



Mathematical model of PSO

$$\overrightarrow{X_i^{t+1}} = \overrightarrow{X_i^t} + \overrightarrow{V_i^{t+1}}$$

$$\overrightarrow{V_i^{t+1}} = w\overrightarrow{V_i^t} + c_1 r_1 \left(\overrightarrow{P_i^t} - \overrightarrow{X_i^t}\right) + c_2 r_2 \left(\overrightarrow{G^t} - \overrightarrow{X_i^t}\right)$$

Inertia

Cognitive component

Social component

Introduction to the PSO: **Concept**

- Uses a number of agents (**particles**) that constitute a swarm moving around in the search space looking for the best solution

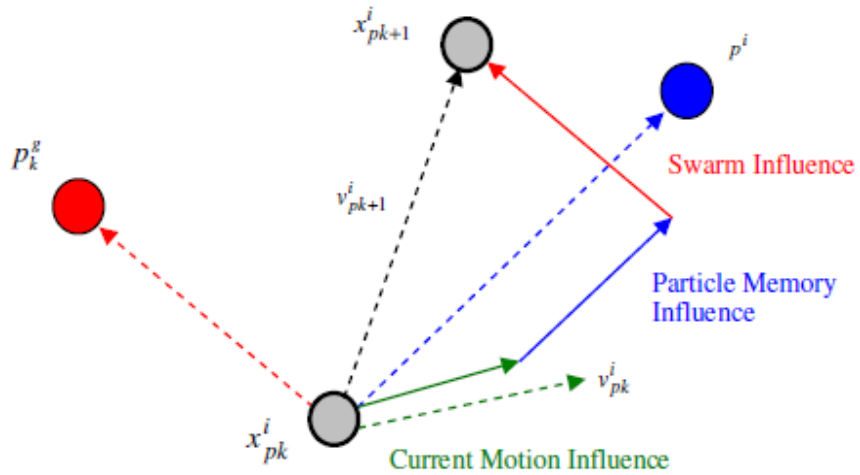
- Each particle in search space adjusts its “flying” according to its own flying experience as well as the flying experience of other particles

Introduction to the PSO: **Concept**

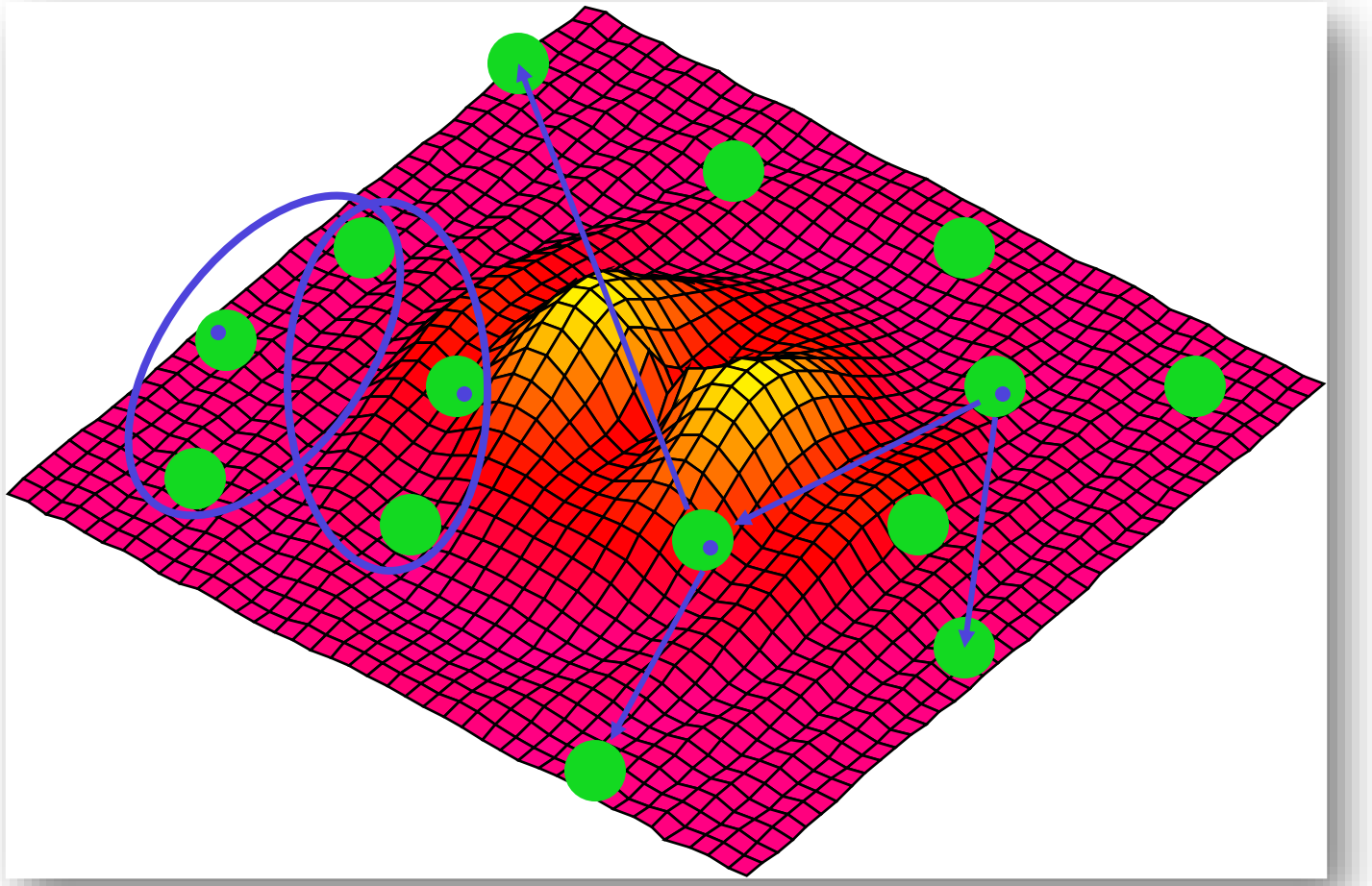
- Collection of flying particles (swarm) - Changing solutions
- Search area - Possible solutions
- Movement towards a promising area to get the global optimum
- Each particle keeps track:
  - its best solution, personal best, *pbest*
  - the best value of any particle, global best, *gbest*

Introduction to the PSO: **Concept**

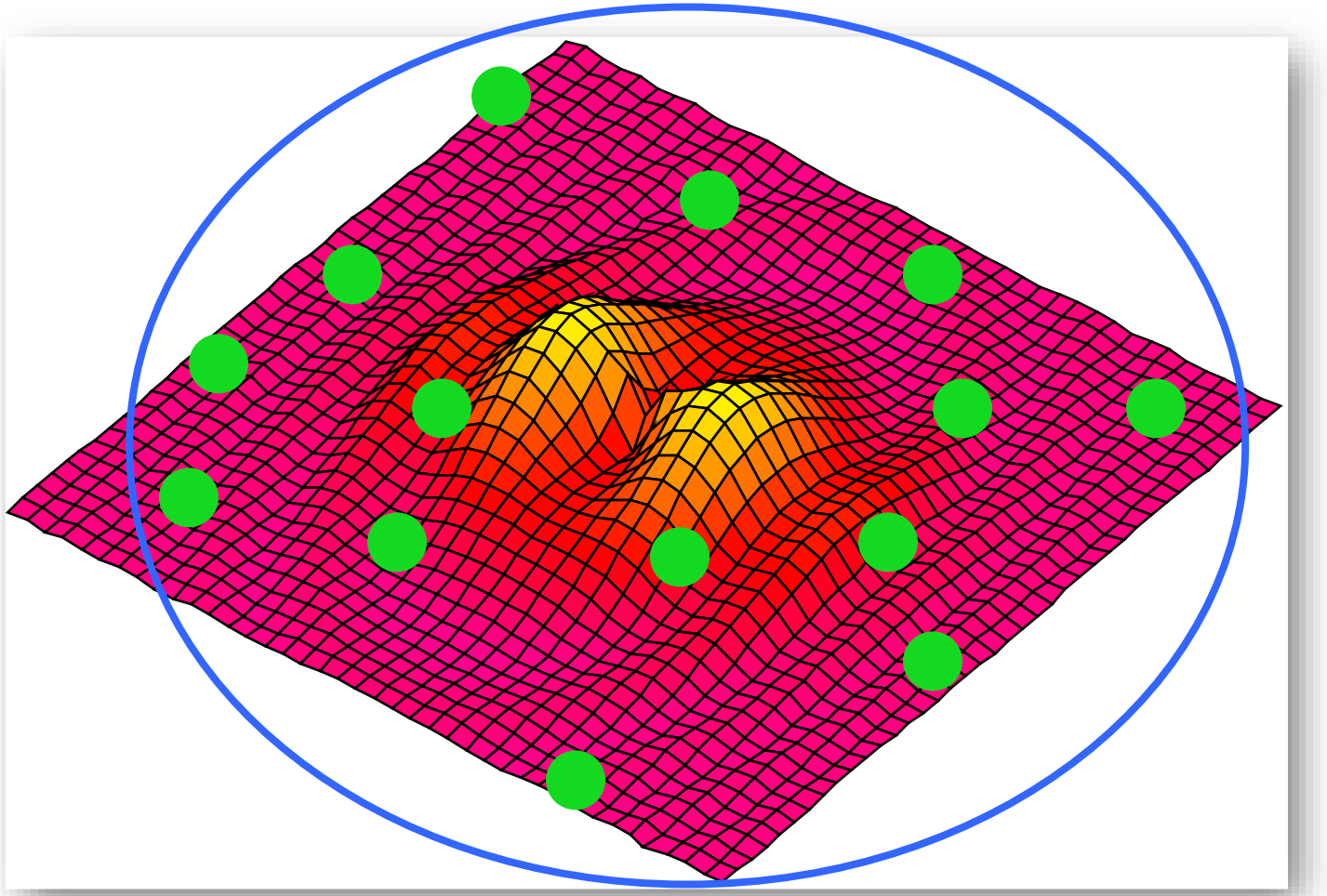
- Each particle modifies its position according to:
  - its current position
  - its current velocity
  - the distance between its current position and *pbest*
  - the distance between its current position and *gbest*



Introduction to the PSO: **Algorithm - Neighborhood**



Introduction to the PSO: **Algorithm - Neighborhood**



Pseudo code of PSO

```
Initialize the controlling parameters (N, c1, c2, Wmin, Wmax, Vmax, and MaxIter)

Initialize the population of N particles

do
    for each particle
        calculate the objective of the particle
        Update PBEST if required
        Update GBEST if required
    end for

    Update the inertia weight
    for each particle
        Update the velocity (V)
        Update the position (X)
    end for

white the end condition is not satisfied

Return GBEST as the best estimation of the global optimum
```

Introduction to the PSO: Algorithm - Parameterss

- Algorithm parameters
  - $\boldsymbol{A}$: Population of agents
  - $\boldsymbol{p_i}$: Position of agent $\boldsymbol{a_i}$ in the solution space
  - $\boldsymbol{f}$: Objective function
  - $\boldsymbol{v_i}$: Velocity of agent's $\boldsymbol{a_i}$
  - $\boldsymbol{V(a_i)}$: Neighborhood of agent a_i (fixed)
- The neighborhood concept in PSO is not the same as the one used in other meta-heuristics search, since in PSO each particle's neighborhood never changes (is fixed)

Introduction to the PSO: **Algorithm**

[x*] = PSO()

P = Particle_Initialization();

For *i*=1 to *it_max*

  For each particle *p* in *P* do

    fp = f(*p*);

    If *fp* is better than f(*pBest*)

      pBest = *p*;

    end

  end

  gBest = best *p* in *P*;

  For each particle *p* in *P* do

    v = w*v* + *c1***rand**(*pBest* – *p*) + *c2***rand**(*gBest* – *p*);

    p = *p* + *v*;

  end

end

Introduction to the PSO: Algorithm

- Particle update rule

$$p = p + v$$

  - with

$$v = wv + c_1 * rand * (pBest - p) + c_2 * rand * (gBest - p)$$

  - Where
  - W is weight

- p: particle's position
- v: path direction
- c_1: weight of local information
- c_2: weight of global information
- *pBest*: best position of the particle
- *gBest*: best position of the swarm
- *rand*: random variable

Introduction to the PSO: Algorithm - Parameters

- Number of particles usually between 10 and 50
- C_1 is the importance of personal best value
- C_2 is the importance of neighborhood best value
- Usually $C_1 + C_2 = 4$ (empirically chosen value)
- If velocity is too low → algorithm too slow
- If velocity is too high → algorithm too unstable

Introduction to the PSO: **Algorithm**

1. Create a ‘population’ of agents (particles) uniformly distributed over X

2. Evaluate each particle’s position according to the objective function

3. If a particle’s current position is better than its previous best position, update it

4. Determine the best particle (according to the particle’s previous best positions)

Introduction to the PSO: **Algorithm**

5. Update particles' velocities:

$$\mathbf{v}_i^{t+1} = \underbrace{\mathbf{v}_i^t}_{inertia} + \underbrace{c_1 \mathbf{U}_1^t (\mathbf{pb}_i^t - \mathbf{p}_i^t)}_{personal\ influence} + \underbrace{c_2 \mathbf{U}_2^t (\mathbf{gb}^t - \mathbf{p}_i^t)}_{social\ influence}$$

6. Move particles to their new positions:

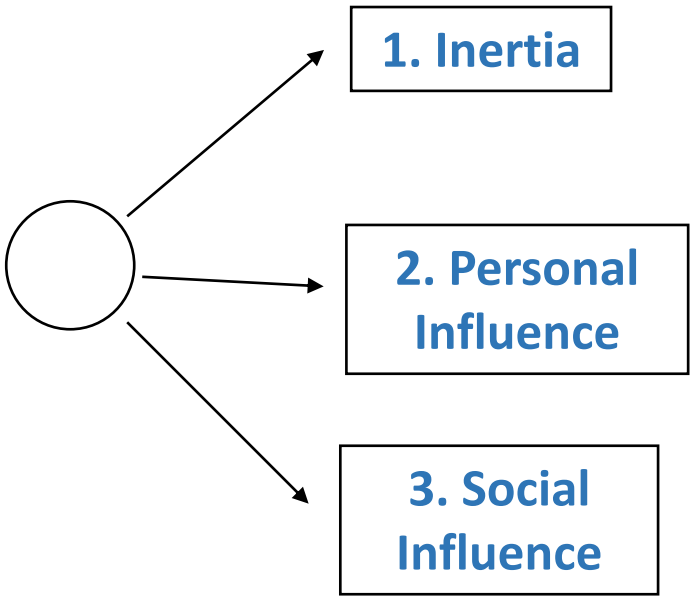
$$\mathbf{p}_i^{t+1} = \mathbf{p}_i^t + \mathbf{v}_i^{t+1}$$

7. Go to step 2 until stopping criteria are satisfied

Introduction to the PSO: Algorithm

Particle's velocity:

$$\mathbf{v}_i^{t+1} = \underbrace{\mathbf{v}_i^t}_{\text{inertia}} + \underbrace{c_1 \mathbf{U}_1^t (\mathbf{pb}_i^t - \mathbf{p}_i^t)}_{\text{personal influence}} + \underbrace{c_2 \mathbf{U}_2^t (\mathbf{gb}^t - \mathbf{p}_i^t)}_{\text{social influence}}$$



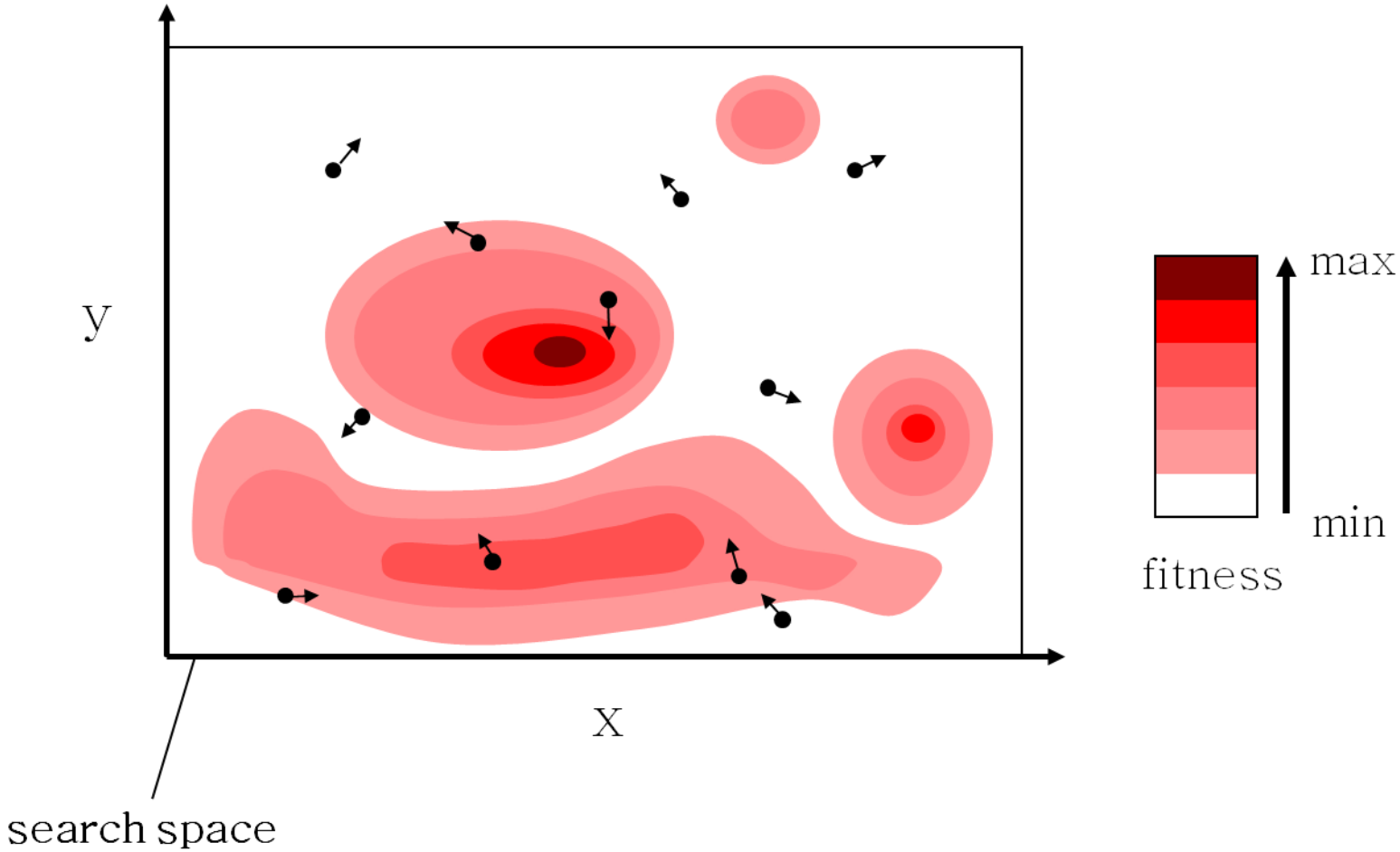
- **Makes the particle move in the same direction and with the same velocity**
- **Improves the individual**
- **Makes the particle return to a previous position, better than the current**
- **Conservative**
- **Makes the particle follow the best neighbors direction**

Introduction to the PSO: Algorithm

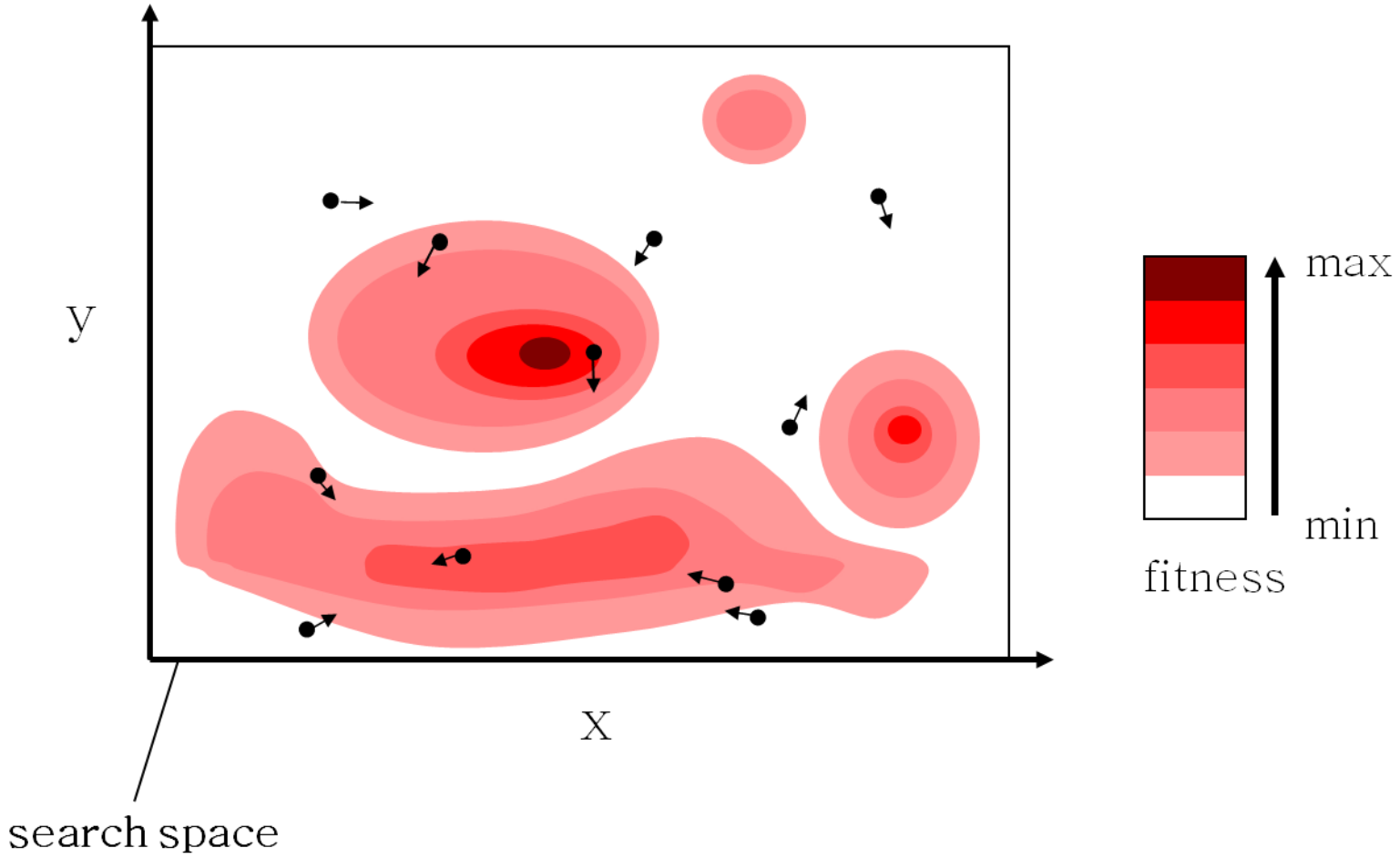
- Intensification: explores the previous solutions, finds the best solution of a given region
- Diversification: searches new solutions, finds the regions with potentially the best solutions
- In PSO:

$$\mathbf{v}_i^{t+1} = \underbrace{\mathbf{v}_i^t}_{\text{Diversification}} + \underbrace{c_1 \mathbf{U}_1^t(\mathbf{pb}_i^t - \mathbf{p}_i^t) + c_2 \mathbf{U}_2^t(\mathbf{gb}^t - \mathbf{p}_i^t)}_{\text{Intensification}}$$

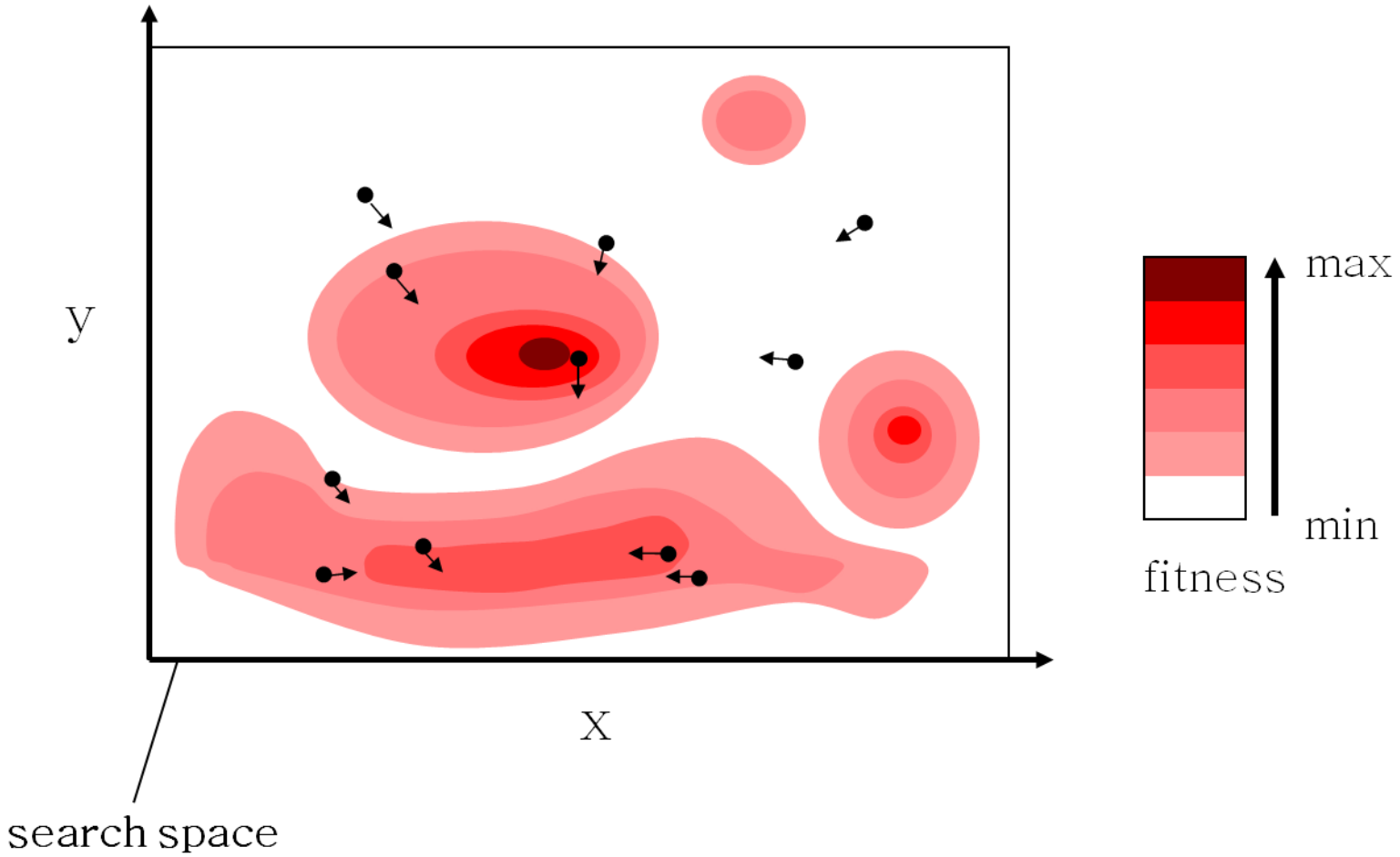
Introduction to the PSO: Algorithm - Example



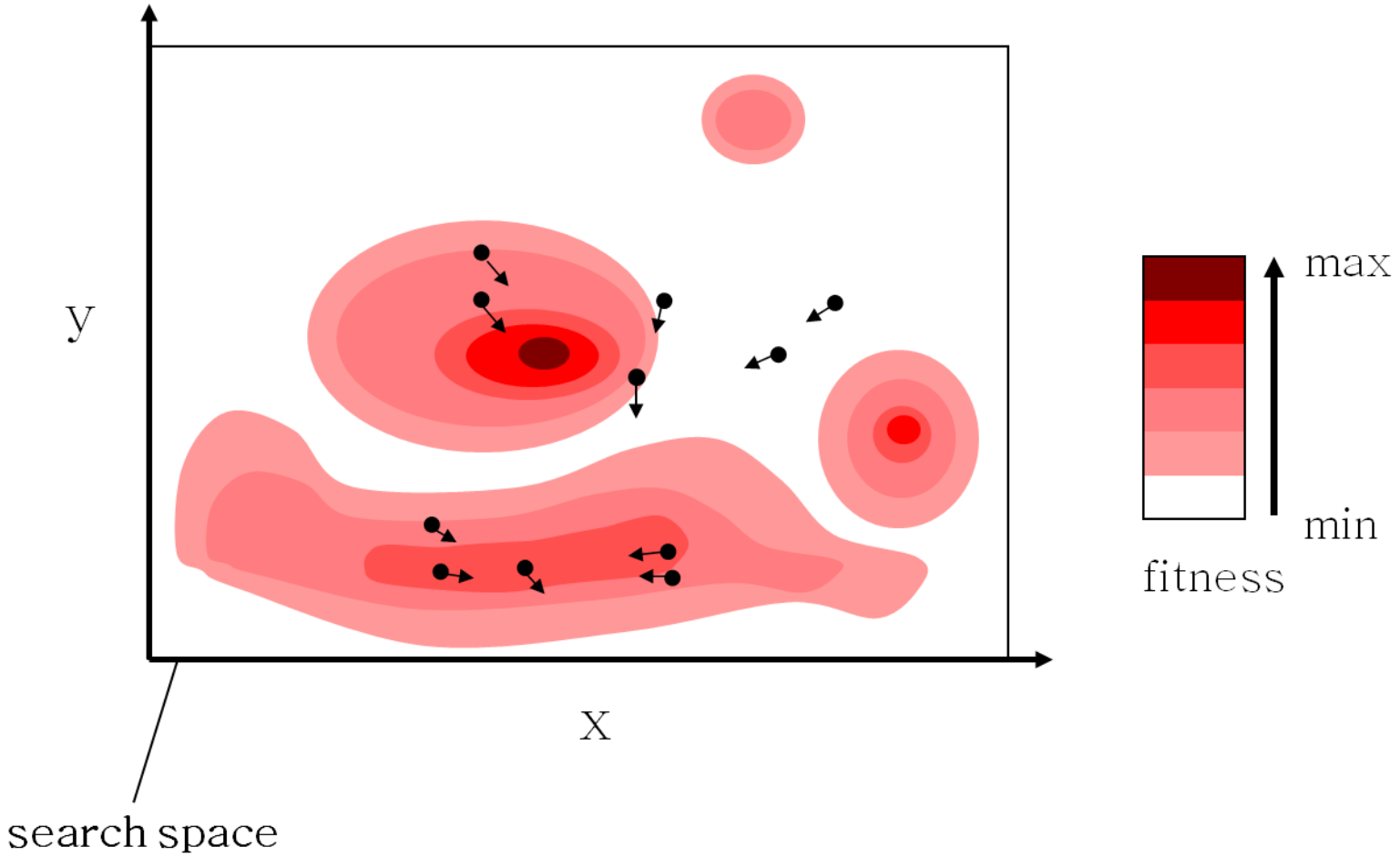
Introduction to the PSO: Algorithm - Example



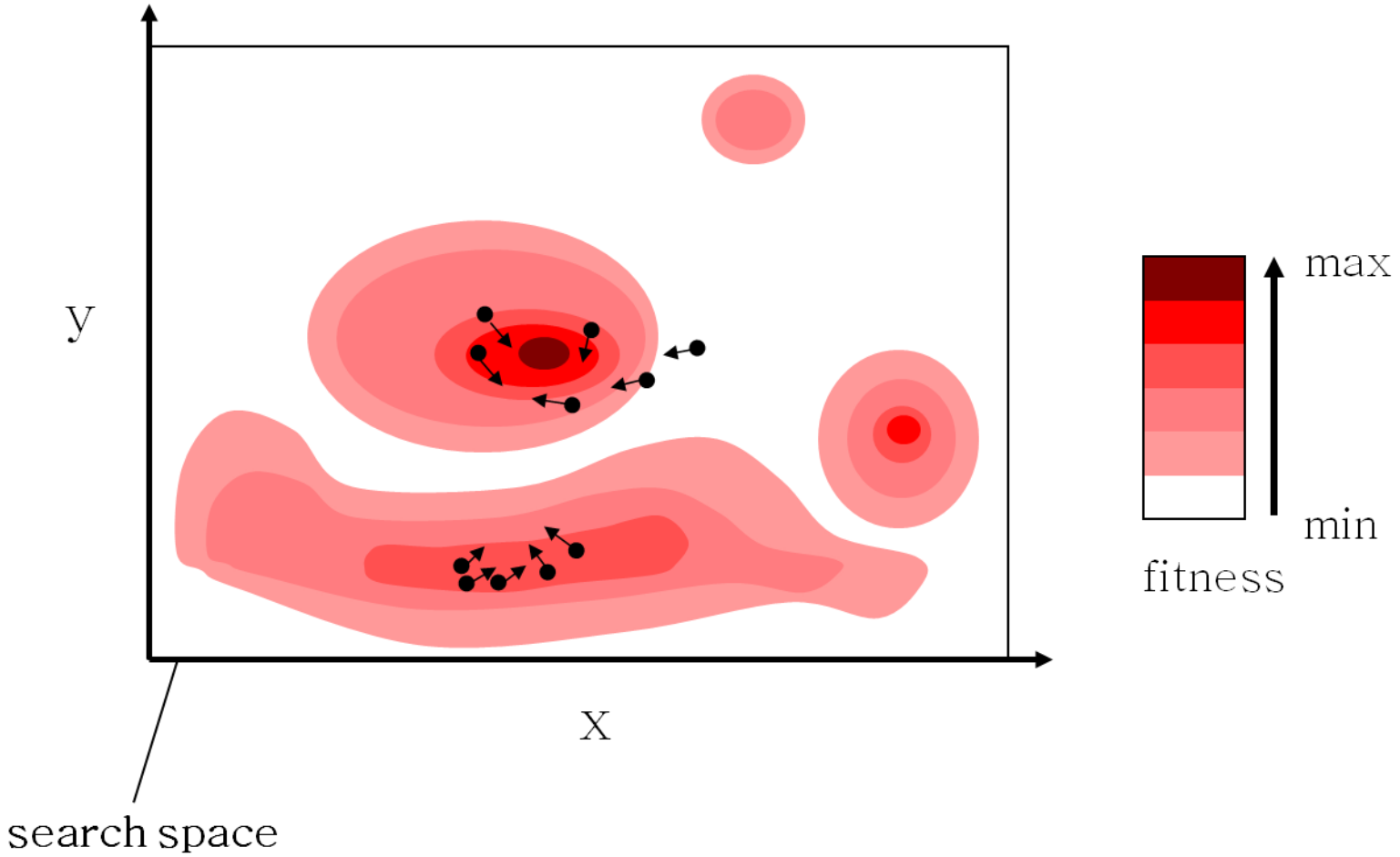
Introduction to the PSO: Algorithm - Example



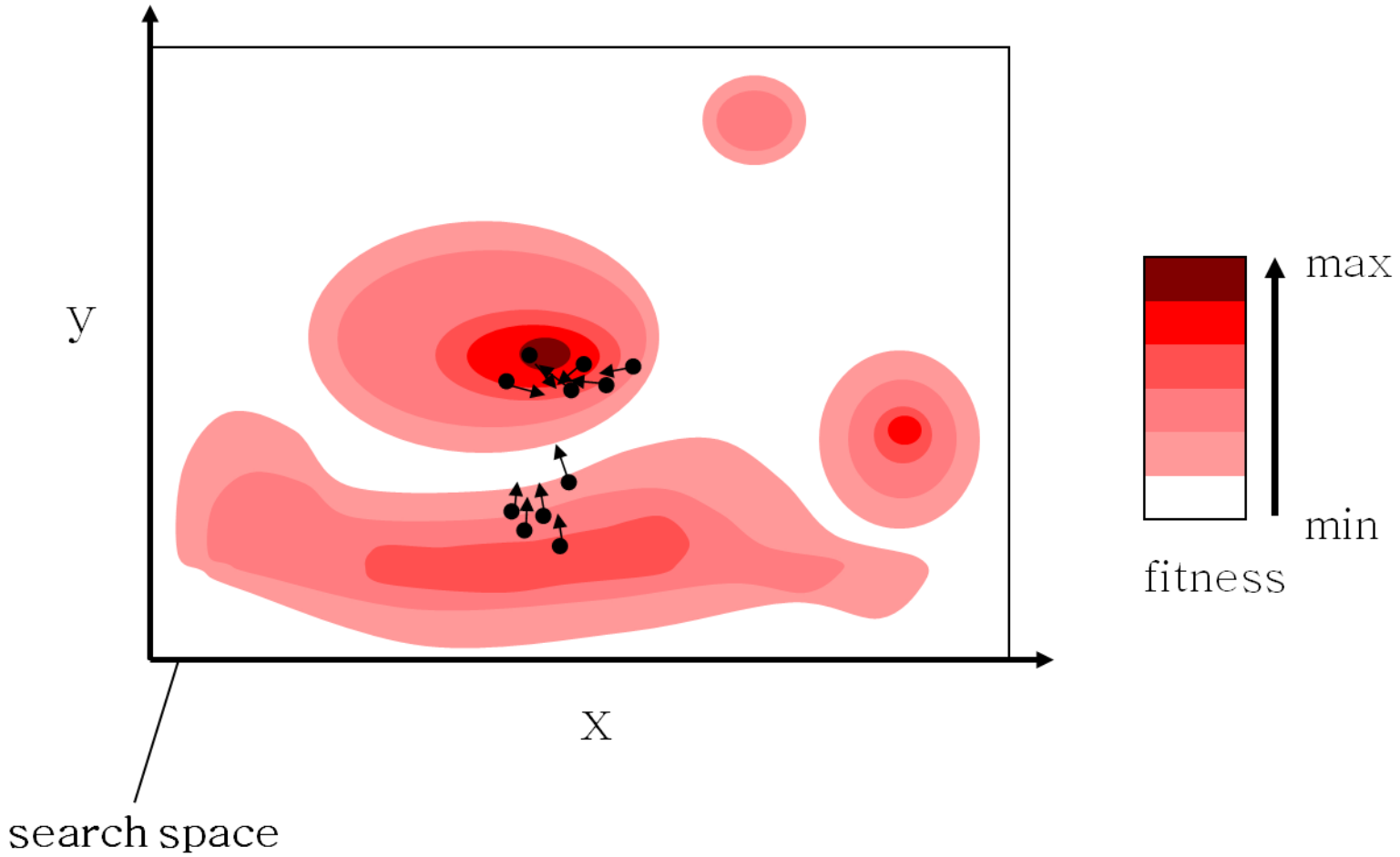
Introduction to the PSO: Algorithm - Example



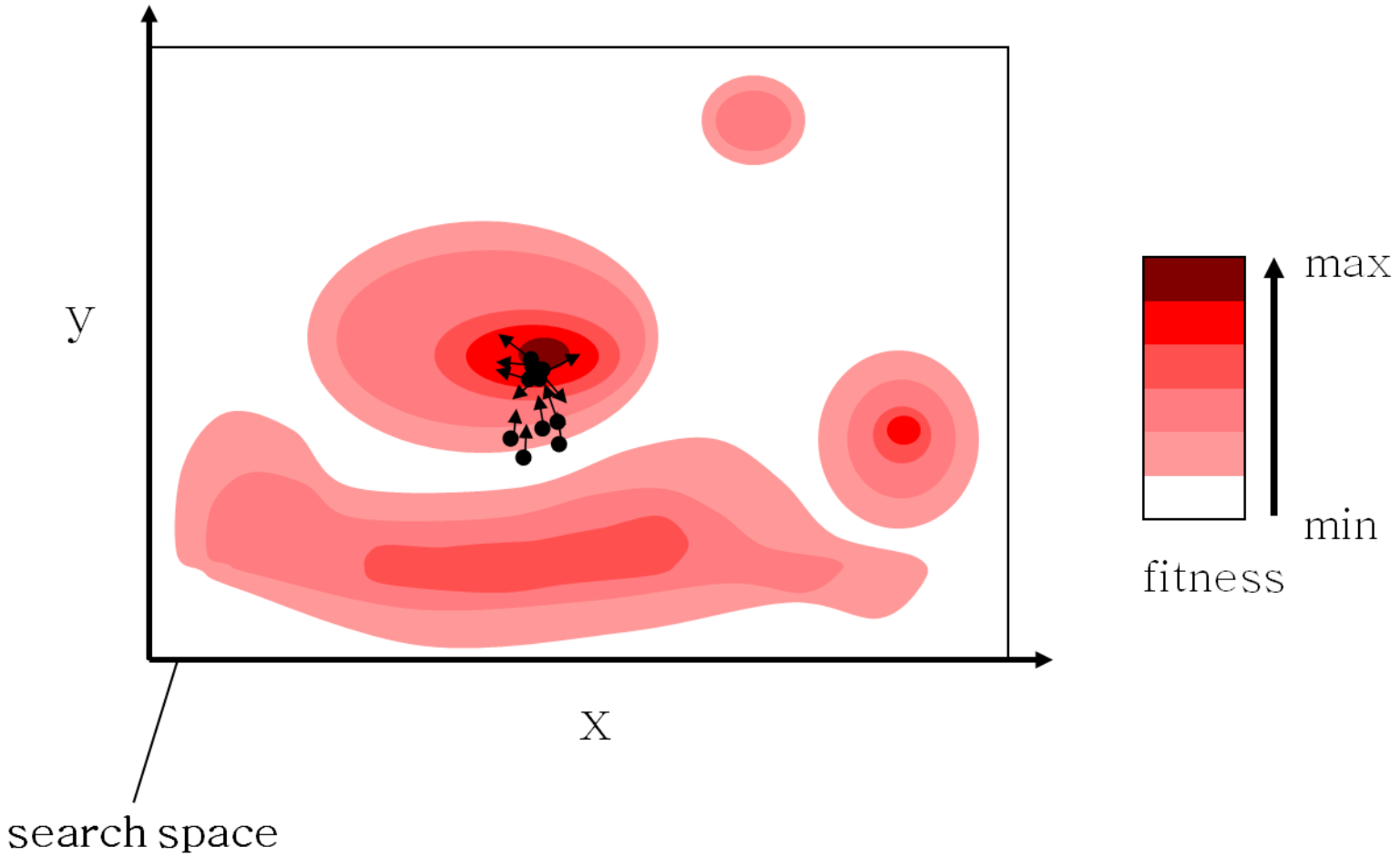
Introduction to the PSO: Algorithm - Example



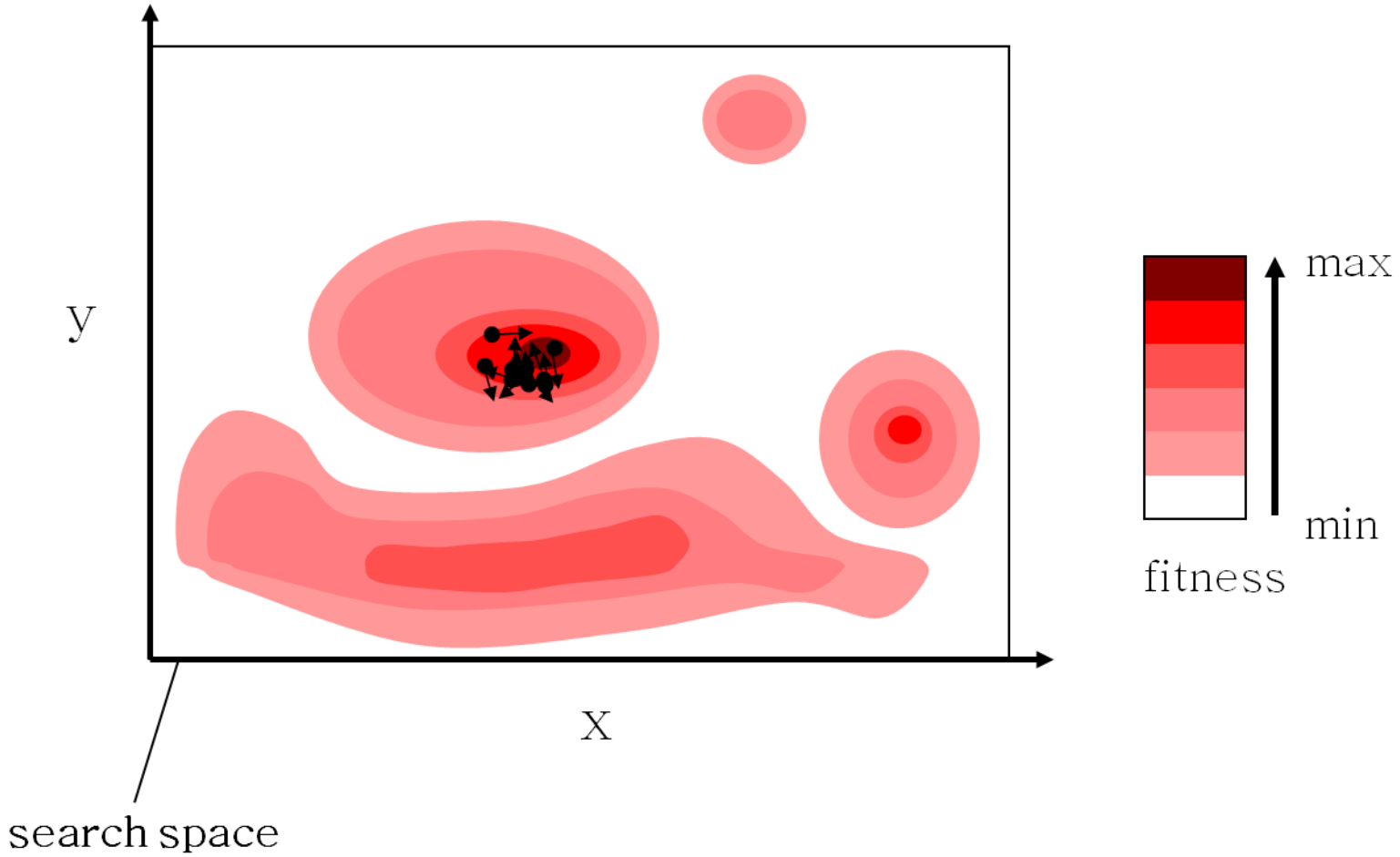
Introduction to the PSO: Algorithm - Example



Introduction to the PSO: Algorithm - Example



Introduction to the PSO: Algorithm - Example



Introduction to the PSO: Algorithm Characteristics

- **Advantages**
  - Insensitive to scaling of design variables
  - Simple implementation
  - Easily parallelized for concurrent processing
  - Derivative free
  - Very few algorithm parameters
  - Very efficient global search algorithm
- **Disadvantages**
  - Tendency to a fast and premature convergence in mid optimum points
  - Slow convergence in refined search stage (weak local search ability)

Solved Example

Consider the optimization problem as follows:

Max $f(x1, x2) = x1^2+x2^2-x1x2+2x1+4x2+3$ **$-5 <= x1, x2 <= 5$**

Parameters of PSO are as follow:

Population size to be 5

c1=c2=1.5

w=0.9

r_1= 0.5949

r_2=0.0855

Maximum number of iteration = 20

Dimension of the problem=2

We initialize the positions of 5 particles randomly using uniform distribution in the range (-5,I5)

$$x = \begin{array}{|cc|l} 3.1472 & -4.0246 & (x_{11}, x_{12}) = x_1 \\ 4.0579 & -2.2150 & (x_{21}, x_{22}) = x_2 \\ -3.7301 & 0.4688 & (x_{31}, x_{32}) = x_3 \\ 4.1338 & 4.5751 & (x_{41}, x_{42}) = x_4 \\ 1.3236 & 4.6489 & (x_{51}, x_{52}) = x_5 \end{array}$$

The corresponding initial fitness function values are

31.9645, 32.6168, 13.2971, 48.6753, 41.4537

Maximum fitness value is 48.6753 and corresponding solution value is [4.1338, 4.5751] is called **gbest**. Also, Since no previous iteration exists, so we call this initial particle as **pbest** $= x$.

Step 2: Select random velocity

Vel=	**0.2194**	**0.2449**	**(v11, v12)**
	0.1908	**0.2228**	**(v21, v22)**
	0.3828	**0.3232**	**(v31, v32)**
	0.3976	**0.3547**	**(v41, v42)**
	0.0934	**0.3773**	**(v51, v52)**

For 1st particle: for 1st components velocity update

$V11 = wv_{11} + c_1 r_1 (pbest_{11} - x_{11}) + c_2 r_2 (gbest_{11} - x_{11})$

= 0.9*0.2194+ (1.5) (0.5949) (3.1472-3.1472) + (1.5) (0.0855) (4.1338-3.1472)

=0.3240

X11=x11+v11

=3.4712 ϵ (-5, 5)

$$
\begin{aligned}
v_{12} &= wv_{12} + c_1 r_1 (pbest_{12} - x_{12}) + c_2 r_2 (gbest_{12} - x_{12}) \\
&= 0.9 \times 0.2449 + (1.5)(0.1174)(-4.0246 + 4.0246) \\
&\qquad + (1.5)(0.0855)(4.5751 + 4.0246) \\
&= 9.6414 \\
x_{12} &= x_{12} + v_{12} \\
&= 5.6168 \notin (-5, 5). Thus, x_{12} = 5
\end{aligned}
$$

Therefore, 1st particle after PSO update becomes

$$x_1 = (3.4712, 5.0000)$$

For 2nd particle:

$$\begin{aligned} v_{21} &= (0.9)(0.1908) + (0.3933)(4.0579 - 4.0579) \\ &\quad + (0.3937)(4.1338 - 4.0579) \\ &= 0.2016 \\ x_{21} &= 4.0579 + 0.2016 \\ &= 4.2595 \in (-5, 5) \end{aligned}$$

Similarly, $x_{22} = 2.9620$. Therefore, 2nd particle after PSO update becomes

$$x_2 = (4.2595, 2.9620)$$

Hence, updated swarms's are $x =$

3.4712	5.0000	$(x_{11}, x_{12}) = x_1$
4.2595	2.9620	$(x_{21}, x_{22}) = x_2$
5.0000	4.3231	$(x_{31}, x_{32}) = x_3$
4.4916	4.8943	$(x_{41}, x_{42}) = x_4$
5.0000	4.9377	$(x_{51}, x_{52}) = x_5$

and their fitness vector are

$$49.6357, \quad 37.6669, \quad 52.3660, \quad 53.7062, \quad \mathbf{57.4433}$$

Since Maximum is 57.4433 and its position is x_5. Therefore, $gbest = x_5 = (5.0000, 4.9377) = (gbest_{i1}, gbest_{i2})$

For 1st particle.
Fitness in previous swarm = 31.9645
Fitness in current swarm = 49.6357
Since fitness in current swarm > fitness in previous swarm, so we set $pbest_1 = x_1 = (3.4712, 5.0000)$
For other particles, we have

$$pbest_2 = x_2 = (4.2595, 2.9620)$$
$$pbest_2 = x_3 = (5.0000, 4.3231)$$
$$pbest_2 = x_4 = (4.4916, 4.8943)$$
$$pbest_2 = x_5 = (5.0000, 4.9377)$$