Computational Design and Fabrication of Hanging Structures

Supplemental Material

Boyu Song
Peking University
boyusong@pku.edu.cn

Jie Feng
Peking University
feng_jie@pku.edu.cn

Bingfeng Zhou
Peking University
cczbf@pku.edu.cn

1 VARIABLE DEFINITIONS

In this section we list all variables and their definitions in the paper (Table.1). These symbols are generic in both the description of our method and the pesudocode below (Alg.1).

Symbol	Definition
N	number of rods
m_i	mass of rod i
l_i	length of rod i
$\vec{M}$	the N-dimensional mass vector of all rods
$\vec{L}$	the N-dimensional length vector of all rods
θ_i	the angles of the rod i relative to the horizontal plane
$\vec{\Theta}$	the N-dimensional angle vector of all rods
$\phi_{i,i+1}$	the angle between rod i and rod $i+1$
u	number of geometry parameters
$\vec{G}$	the geometry parameter vector of all rods
p_{Ti}	the i^{th} junction of target shape T
p_{Si}	the i^{th} junction of simulation shape S
γ_j	particle j in PSO algorithm
γ_j^k	the position of particle γ_j in the k^{th} iteration, which is a configuration vector
Γ_j^k	a angle vector, i.e. the simulation result of a configuration P_j^k
$\widetilde{D}$	the Euclidean distance between two curve
$\vec{D}_j^k$	the weighted sum of the deviation vector, i.e. the directed distance between the junction points
γ_{lj}^k	the *local best* position of particle j in the k^{th} iteration
γ_g^k	the *global best* position of whole swarm in the k^{th} iteration
γ_{bj}^k	the *better* position for particle j in the k^{th} iteration
γ_{wj}^k	the *worse* position for particle j in the k^{th} iteration

Table 1: Variable definitions

2 THE IMPROVED PSO ALGORITHM

Given a user-specified shape $\vec{\Theta}$, our configuration optimization algorithm will find an optimal configuration $(\vec{M}, \vec{G})$ to form the target shape.

We solve this non-linear problem by a improved-particle swarm optimization(PSO) algorithm. Compared with the original PSO algorithm, we apply two more guide points called the *better solution* point and the *worse solution* point, while keeping the two original points, i.e. the *local best* point and the *global best* point. Alg.1 shows the pseudocode of our algorithm.

Here function *simulateShape*() is the shape simulation algorithm given in Chapter4.1 in the paper.

Function *selectionPrinciple*() serve to provide the *better* and *worse* position for current particle according to deviation directions $\vec{D}$. In Fig.1, yellow arrow indicates the deviation direction bwtween target shape(green) and the current simulation shape(red), which is applied as the error descriptor in our work. The direction vector is the sum of all blue vectors.

The selection principles have been described in the paper, but for getting a clearer understanding of the principles, we list several typical cases in the form of diagrams(Fig.2 and Fig.3).

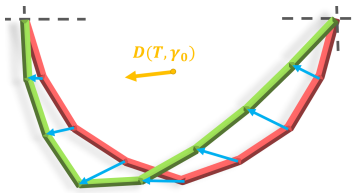

Figure 1: Deviation direction.

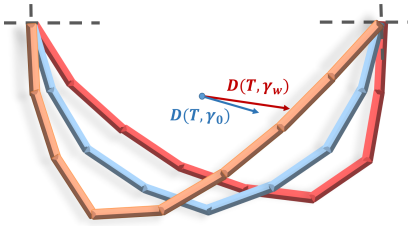

Figure 2: A typical *worse* condition. Here, the target shape is shown in orange. The shape in red is a *worse* shape than current one (blue).

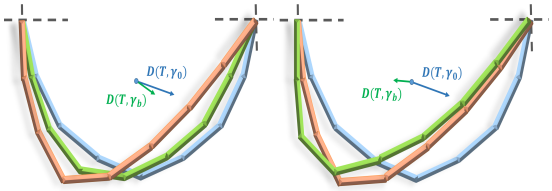

Figure 3: Two *better* conditions. Here, the shapes in green are *better* shape than current one (blue).

3 ALGORITHM PERFORMANCE

Table2 demonstrate the performance of our improved PSO algorithm in some typical cases. We can see that the calculation cost is directly related to the complexity of target structures. The polygon configuration optimization cost more time before converging than Semi-circle and triangle. This is because the optimal solution is almost located on the boundary of search-space, so the particles can hardly know the information of the optimal solution. But over all, the time-consuming of our method is acceptable.

Target shape	Number of rods	Iteration number	Consuming-time(s)
Semi-circle	8	7	6.217
	12	17	37.251
	20	21	143.171
Triangle	8	8	8.145
	12	14	28.805
	20	16	90.704
Polygon	8	16	38.585
	12	26	96.581
	20	29	246.799

Table 2: Configuration Optimization Performance

Algorithm 1 Configuration optimization

Input: The vector of angles $\vec{\Theta}$;
Output: The vector of mass $\vec{M}$; The vector of geometry constrains $\vec{G}$.

// initilization
$k \leftarrow 1$
$\widetilde{D}_{gb} \leftarrow$DBL_MAX
for each particle γ_i $i \in [1, m]$ **do**
  Initialize particle's position $\gamma_i = (M_i^0, G_i^0)$ randomly
  Initialize particle's local best position $\gamma_{li}^0 = \gamma_i^0$ randomly
  $\Gamma_i^0 \leftarrow simulateShape(\gamma_i^0)$
  $\widetilde{D}_i^0 \leftarrow \widetilde{D}(T, \Gamma_i^0)$
  if $\widetilde{D}_i^0 < \widetilde{D}_g$ **then**
    Update global best position $\gamma_g \leftarrow \gamma_i^0$
  end if
  Initialize particle's velocity V_i^0 randomly
  $\vec{D}_i^0 \leftarrow \vec{D}(T, \Gamma_i^0)$
end for
for each particle γ_i $i \in [1, m]$ **do**
  $[\gamma_{bi}^0, \gamma_{wi}^0] \leftarrow selectionPrinciple(\vec{D^0})$
end for
// iteration
while k < *max-Iteration-Number* **or** $\widetilde{D}(T, \Gamma_{gb}) > \epsilon$ **do**
  // Update velocity and location of particles
  for each particle γ_i $i \in [1, m]$ **do**
    $V_i^k \leftarrow V_i^{k-1} + rand()(\gamma_{li}^{k-1} - \gamma_i^{k-1})$
    $V_i^k \leftarrow V_i^k + rand()(\gamma_g^{k-1} - \gamma_i^{k-1})$
    $V_i^k \leftarrow V_i^k + rand()(\gamma_{bi}^{k-1} - \gamma_i^{k-1})$
    $V_i^k \leftarrow V_i^k + rand()(\gamma_{wi}^{k-1} - \gamma_i^{k-1})$
    $P_i^k \leftarrow \gamma_i^{k-1} + V_i^k$
  end for
  // update fitness variables
  for each particle γ_i $i \in [1, m]$ **do**
    $\Gamma_i^k \leftarrow simulateShape(\gamma_i^k)$
    $\widetilde{D}_i^k \leftarrow \widetilde{D}(T, \Gamma_i^k)$
    $\vec{D}_i^k \leftarrow \vec{D}(T, \Gamma_i^k)$
  end for
  for each particle γ_i $i \in [1, m]$ **do**
    if $\widetilde{D}(T, \Gamma_i^k) < \widetilde{D}(T, \Gamma_{li}^{k-1})$ **then**
      Update local best position $\gamma_{li}^k \leftarrow \gamma_i^k$
    end if
    if $\widetilde{D}(T, \Gamma_i^k) < \widetilde{D}(T, \Gamma_{gb})$ **then**
      Update global best position $\gamma_{gb} \leftarrow \gamma_i^k$
    end if
  end for
  for each particle γ_i $i \in [1, m]$ **do**
    $[\gamma_{bi}^0, \gamma_{wi}^0] \leftarrow selectionPrinciple(\vec{D^k})$
  end for
  $k \leftarrow k + 1$
end while