Chapter Three

V-shaped beam design and simulations

This chapter, discuss V-shaped beam design and simulations develop from steady state heat equation followed by heat dissipation (entropy) govern formula to generate mathematical model for each parameter to implement PSO technique by MATLAB codes we take some consideration to implement the method during this chapter.

3. V-shaped beam design and simulations:

Figure (3.1) shows the schematic drawing of V-shaped beam actuator of High aspect ratio structure & (b) V-shaped beam deformed in the arched direction owing to a dc bias to reach the optimized coupling. The main considerations of designing appropriate V-beam actuators are force output and displacement. Cochran et al. (Cochran et al. 2003; Maloney et al.2004) have developed a model to derive the maximum force output, denoted as the Ft, from a single V-beam actuator. Assuming thermal conductivity and theCoefficient of thermal expansion are temperature independent, the steady state heat equation is**[2]:**

$$k_s \frac{dT(x)}{dx^2} + J^2\rho - Sk_a \frac{T(x) - 273}{gh} = 0 \qquad (3.1)$$

In present case, the C is the experimental correlation factor regarding to excessive deflection based on measured data and is set as 1.5, and the up-limit of silicon temperature caused by joule heating is set as 800 K in the center of V-beam actuator .Solving equation (1) for temperature will derive,

$$T(x) = 273 + \frac{J^2\rho}{k_p gh}\left[1 - \frac{\cosh\left(^{mL}/_{2}\right) - mx}{\cosh\left(^{mL}/_{2}\right)}\right] \qquad (3.2)$$

$$m = \frac{S\,Ka}{Ks\,gh} \qquad (3.3)$$

$$S = \frac{[2g + h] + w}{w} \qquad (3.4)$$

$$\frac{F_t}{p} = \frac{4\alpha CEd}{k_s m^2 L^2}\left[1 - \frac{2\tanh\left(\frac{mL}{2}\right)}{mL}\right] \quad (3.5)$$

$$P = w\,h\,L\,J^2\rho \quad (3.6)$$

Where T(x) means the temperature variation of V-beam, x means the position along with the beam, P is the electrical power. L, w, h, and d denote V-beam length, width, thickness, and center offset distance, respectively.

Besides, the g is defined as the gap between the bottom of V-beam actuator and the substrate, it is 2 μm. The other parameters listed in Table (3.1) include ρ of the resistivity of silicon, J of the current density, α of the thermal expansion coefficient of silicon, ka and ks of the thermal conductivities of the air and silicon; S (entropy) of the shape parameter which means the ratio of the heat loss from sides and bottom of the V-beam to the heat loss from the bottom of the beam only **[2].**

table(3.1)Parameters used in analytical modeling of V- beam **[2]**

parameter		Value
C	**Experimental correlation factor**	**1.5**
E	**Young's modulus , silicon**	**169 GPA**
Ka	**Thermal conductivity ,air**	**0.026 W/m-K**
Ks	**Thermal conductivity , silicon**	**150 W/m-K**
d	**Density , silicon**	**2330 Kg/m3**
α	**Thermal expansion coefficient ,silicon**	**$2.51 * 10^{-6} K^{-1}$**
ρ	**Resistivity , silicon**	**0.038 Ω-cm**
θ	**tilted angle**	**0.6 degree**

Solving equations (3.4) & (3.6) and subsisted for w we get,

$$S = \frac{[hLJ^2\rho][2g + h] + P}{P}$$

Arrange the above equation we get:

$$S = \frac{[2ghLJ^2\rho] + [h^2LJ^2\rho] + P}{P} \quad (3.7)$$

&solving equation (5) for F_t becomes,

$$F_t = \frac{4P\alpha CEd}{k_s m^2 L^2}\left[1 - \frac{2\tanh\left(\frac{mL}{2}\right)}{mL}\right] \quad (3.8)$$

And for angular displacement

$$U(\theta) = P\left(1 - \frac{2\tanh\frac{(mL)}{2}}{mL}\right)\frac{C\,\alpha\,L\,d(\theta)}{4\,w\,h\,Ks\,m^2\,(d(\theta)^2 + w^2)} \qquad (3.9)$$

Where F_t is output force of V beam &U(θ) (angular displacement) means the relation between displacement andθ, θ means the tilted angle of V-beam to perpendicular direction regarding to moving axis**[2].**

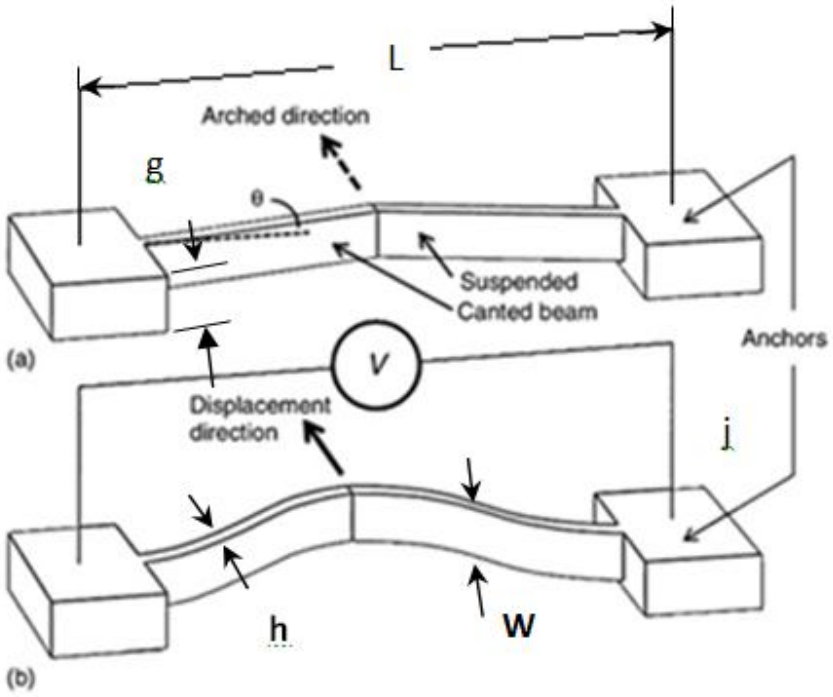

Figure (3.1) (a) Schematic drawings of V-shaped beam actuator of High-aspect ratio structure. (b) V-shaped beam deformed in the Arched direction owing to a dc bias**[45]**

3.1. Mathematical model:

Take the design parameters are:

h = beam thickness = x_1

g = gap between beam and substrate = x_2

w = V beam width = x_3

L = V beam length = x_4

J = current density = x_5

And the design boundaries corresponding to each design parameter are:

50 GPa $\leq x_1 \leq$ 140 GPa	step = 10 mean = 95
1 μm $\leq x_2 \leq$ 10 μm	step = 1 mean = 5
2 μm $\leq x_3 \leq$ 20 μm	step = 2 mean = 11
500 μm $\leq x_4 \leq$ 4550 μm	step = 450 mean = 2525
0.1 A/μm^2 $\leq x_5 \leq$ 1 A/μm^2	step = 0.1 mean = 0.5

Subsisted for parameters & value of ρ Equation (7) becomes:

$$S = \frac{[0.00076x_1x_2x_4x_5^2][0.00038x_1^2x_4x_5^2] + P}{P}$$

And equation (6) respectively:

$$P = 0.00038x_1x_3x_4x_5^2$$

We well study any parameter individually from other parameters which are remain constant, we wear approximate the mean value for each parameter and simulate the result s by MATLAB using effective optimizer (PSO) and we may calculate such entropy under various geometrical combination of parameters according to the required design input from PSO consideration.. On the other hand, we may simply conduct MATLAB simulation for determination of an optimized geometrical micromechanism structure for V-shaped beam devices. Several run will carry out for parameters to get the optimized solution.

For Parameterx_1 subsisted the mean values of other parameters:

$$s = \frac{2.39875x_1 + 0.239875x_1^2 + p}{p} \qquad (3.10)$$

$$p = 2.638625x_1 \quad (3.11)$$

For Parameter$x_2$subsisted the mean values of other parameters:

$$s = \frac{45.57625x_2 + 2164.871875 + p}{p} \quad (3.12)$$

$$p = 250.6693575$$

For Parameter$x_3$subsisted the mean values of other parameters:

$$s = \frac{2392.753125 + p}{p} \quad (3.13)$$

$$p = 22.788125x_3 \quad (3.14)$$

For Parameter$x_4$subsisted the mean values of other parameters:

$$s = \frac{0.947625x_4 + p}{p} \quad (3.15)$$

$$p = 0.099275x_4 \quad (3.16)$$

For Parameter$x_5$subsisted the mean values of other parameters:

$$s = \frac{9571.0125x_5^2 + p}{p} \quad (3.17)$$

$$p = 1002.6775x_5^2 \quad (3.18)$$

3.2. PSO consideration:

3.2.1. Parameterx_1:

Number of particle = 10

Particles Current (initial) position for parameter X_1 :

50, 60, 70, 80, 90, 100, 110, 120, 130, 140

Particles current (initial) velocity for parameter V_1 =0

Assume random numbers: rand1=0.5, rand2=0.4

Cognitive learning factor =1

Social learning factor = 2

Iterations =100

$$w = \frac{\text{max iteration} - \text{current iteration}}{\text{max iteration}}$$

3.2.2. Parameterx_2:

Number of particle = 10

Particles Current (initial) position for parameter X_2 :

1, 2,3,4,5,6,7,8,9,10

Particles current (initial) velocity for parameter V_2 =0

Assume random numbers: rand1=0.5, rand2=0.4

Cognitive learning factor =1

Social learning factor = 2

Iterations =100

$$w = \frac{\text{max iteration} - \text{current iteration}}{\text{max iteration}}$$

3.2.3. Parameter x_3:

Number of particle = 10

Particles Current (initial) position for parameter X_3 :

2,4,6,8,10,12,14,16,18,20

Particles current (initial) velocity for parameter V_3 =0

Assume random numbers: rand1=0.5, rand2=0.4

Cognitive learning factor =1

Social learning factor = 2

Iterations =100

$$w = \frac{\text{max iteration} - \text{current iteration}}{\text{max iteration}}$$

3.2.4. Parameterx_4:

Number of particle = 10

Particles Current (initial) position for parameter X_4 :

500, 950, 1400, 1850, 2300, 2750, 3200, 3650, 4100, 4550

Particles current (initial) velocity for parameter V_4 =0

Assume random numbers: rand1=0.5, rand2=0.4

Cognitive learning factor =1

Social learning factor = 2

Iterations =100

$$w = \frac{\max \text{iteration} - \text{current iteration}}{\max \text{iteration}}$$

3.2.5. Parameterx_5:

Number of particle = 10

Particles Current (initial) position for parameter X_5 :

0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1

Particles current (initial) velocity for parameter V_5 =0

Assume random numbers: rand1=0.5, rand2=0.4

Cognitive learning factor =1

Social learning factor = 2

Iterations =100

3.3. MATLAB code:

MATLAB is a commonly used program for computer modeling. Its code is relatively straightforward. So even though you may not use MATLAB, it has a pseudo code flavor that should be easy to translate into your favorite programming language **[46].**The main consideration when we simulate the results,particlesmay occasionally fly to a position beyond the defined search space and generate an invalid solution, either (1) if it passed over and (0) if it passed under the limit to keep the boundary condition.

3.3.1. PSO implementation:

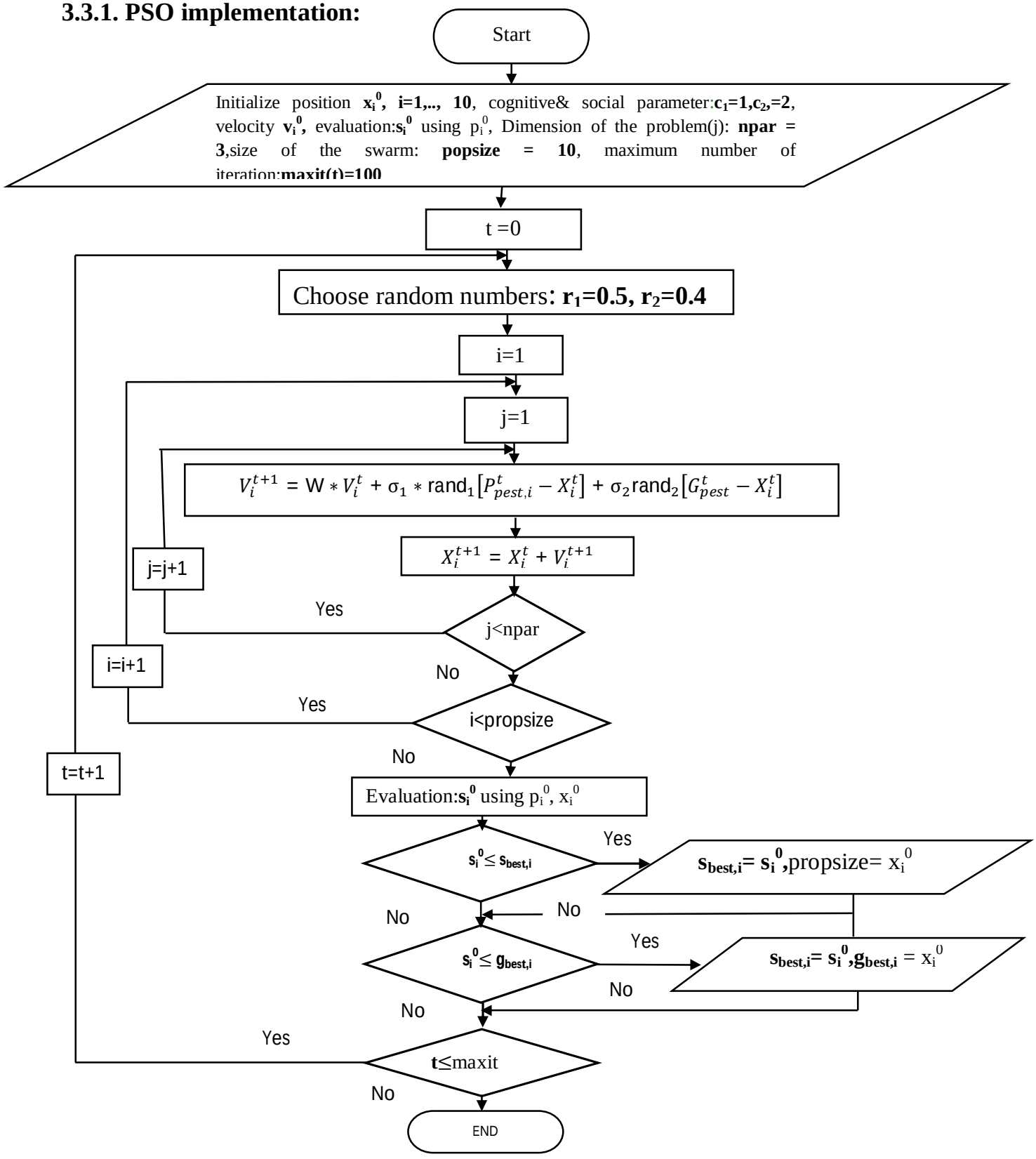

Figure (3.2) PSO implementation by MATLAB program flow chart

```
clc
clear all
ff = 'tut3'; % Objective Function
% Initializing variables
popsize = 10; % Size of the swarm
npar = 3; % Dimension of the problem
maxit = 100; % Maximum number of iterations
c1 = 1; % cognitive parameter
c2 = 2; % social parameter
% Initializing swarm and velocities
xi=50:10:140; %funnum==1 %F1
%xi=1:1:10;    %funnum==2 %F2
%xi=2:2:20;      %funnum==3 %F3
%xi=500:450:4550;  %funnum==4 %F4
%xi=0.1:0.1:1;     %funnum==5 %F5
par=xi'; % random population of
vv=0*xi;
vel = vv' ;% random velocities
cost=feval(ff,par);% calculates population cost
using ff
minc(1)=min(cost); % min cost
globalmin=minc(1) % initialize global minimum
% Initialize local minimum for each particle
localpar = par % location of local minima
localcost = cost % cost of local minima
% Finding best particle in initial population
[globalcost,indx] = min(cost)
globalpar=par(indx,:)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%
% Start iterations
iter = 0; % counter
par_xx([1:10])=xi';
r=1;
vel_xx([1:10])= vel';
```

```
cost_xx([1:10])= cost';
%tplot=[];
whileiter<maxit
iter = iter + 1;
%tplot(iter) =iter;
% update velocity = vel
w=(maxit-iter)/maxit; %inertia weiindxht
r1 = 0.5; % random numbers
r2 = 0.4; % random numbers
vel=(w*vel + c1 *r1.*(localpar-
par)+c2*r2.*(globalpar-par));
% update particle positions
%if vel == 0
   % par =1;
%else
par = par + vel; % updates particle position
%par_xx([r*10+1:10*iter])=par';
%end
overlimit=par<=140;     %funnum==1 %F1
underlimit=par>=50;     %funnum==1 %F1
%overlimit=par<=10;     %funnum==2 %F2
%underlimit=par>=1;     %funnum==2 %F2
%overlimit=par<=20;     %funnum==3 %F3
%underlimit=par>=2;     %funnum==3 %F3
%overlimit=par<=4550;   %funnum==4 %F4
%underlimit=par>=500;   %funnum==4 %F4
%overlimit=par<=1;      %funnum==5 %F5
%underlimit=par>=0.1;   %funnum==5 %F5
par=par.*overlimit+not(overlimit);
par=par.*underlimit;
%%
%%
% Evaluate the new swarm
%clear cost
cost = feval(ff,par) % evaluates cost of swarm
```

```
% Updating the best local position for each
particle
bettercost = cost <localcost;
localcost =
localcost.*not(bettercost)+cost.*bettercost;
globalcost;
localpar(find(bettercost),:)=
par(find(bettercost),:);
% Updating index g
[temp, t] = min(localcost);
if temp<globalcost
globalpar=par(t,:); indx=t; globalcost=temp;
end
print_out=[iterglobalparglobalcost] % print
output each
% iteration
minc(iter+1)=min(cost); % min for this
% iteration
globalmin(iter+1)=globalcost; % best min so far
meanc(iter+1)=mean(cost) ;% avg. cost for
% this iteration
%%
par_xx([r*10+1:10*(iter+1)])=par';
vel_xx([r*10+1:10*(iter+1)])=vel;
cost_xx([r*10+1:10*(iter+1)])=cost;
 r=r+1;
end% while
par_xx=par_xx';
vel_xx=vel_xx';
cost_xx=cost_xx';
xx= [par_xxvel_xxcost_xx];
plot([1:1010]/10,par_xx)% you need to change the
dimension of [1:1010] depending on your
iterations
%axis([0 120 0 40])
```

```
xlabel('iterations')
ylabel('beam thickness (h)um')
%ylabel('gap between beam & substrate (g)um')
%ylabel('V beam width (w)um')
%ylabel('V beam length (L)um')
%ylabel('current density (J)A/um^2 ')
```

3.3.2.Test function:

```
function f=tut3(x)
funnum=1;
iffunnum==1 %F1
%x=50:10:140;
p=2.638625.*x;
f=(2.39875.*x+0.239875.*x.^2+p)./p;
elseiffunnum==2 %F2
%x=1:10;
p=250.6693575;
f=(45.57625.*x+2164.871875+p)./p;
elseiffunnum==3 %F3
%x=2:2:20;
p=22.788125.*x;
f=(2392.753125+p)./p;
elseiffunnum==4 %F4
%x=500:450:4550;
p=0.099275.*x;
f=(0.947625.*x+p)./p;
elseiffunnum==5 %F5
%x=0.1:0.1:1;
p=1002.6775.*x.^2;
f=(9571.0125.*x.^2+p)./p;
end
```