浙江大学

ZJUI 学院 2024 年暑期科研项目总结材料

中文论文题目： 热致变性硅胶设计

英文论文题目： **Thermal deformation silicone design**

成员姓名：王安 （3230110061）

龚钟睿（3230110740）

黄柯衢（3230110348）

指导教师：王冠云

指导教师所在学院：浙江大学计算机学院

暑期研究起止日期：2024 年 06 月 15 日-2024 年 07 月 15 日

摘要

借助膨胀硅胶在加热后膨胀的特性，将其与普通硅胶结合，可衍生出多种应用。本论文主要在三方面进行探究：

1. 膨胀硅胶的基础参数对于膨胀效果的影响
2. 气动硅胶臂的制作工艺优化
3. 基于夹层圆盘的双稳态结构探索。

在进行了多组控制变量膨胀硅胶的测定后，我们发现膨胀硅胶的弯曲程度只与宽度和浓度成正相关，厚度存在一个点使得弯曲程度达到最大。同理，实验显示其产生的应力与浓度和宽度成正相关，厚度存在一个点使得应力达到峰值。

对于气动硅胶臂的制造问题，具体表现为气腔大小不均一、气密性差。我们优化了制作工艺，解决了该问题。原有的办法是分别制造气薄模和主体，再将两部分用硅胶粘合起来。新的方法是先制造主体，并在主体侧面预留栅格状空隙。把主体倒置入模具，一体铸成硅胶薄膜，保证了气密性。预留的空隙可以让硅胶液体充分流动，达到气腔大小均一的效果。

双稳态结构的探索中，不断调整夹层圆盘中心圆的大小、周围齿数、齿面积占比和上下齿交错排列，最终实现了较好的双稳态响应。

Abstract

With the help of expansion silicone's property of expanding when heated, a variety of applications can be derived by combining it with ordinary silicone. In this thesis, we mainly investigate in three aspects: 1. the influence of the basic parameters of the expanded silica gel on the expansion effect 2. the optimization of the fabrication process of the pneumatic silica arm 3. the exploration of the bistable structure based on the sandwiched disc.

After carrying out several sets of measurements of the control variables of the expanded silica gel, we found that the degree of bending of the expanded silica gel is only positively correlated with the width and concentration, and that there exists a point in the thickness that makes the degree of bending reach its maximum. Similarly, experiments show that the stress generated is positively correlated with concentration and width, and that there is a point in thickness where the stress peaks. For the manufacturing problem of pneumatic silicone arm, it is specifically manifested in the uneven size of air cavity and poor air tightness. We optimized the fabrication process to solve the problem. The original approach was to fabricate the air-thin mold and the main body separately, and then bond the two parts together with silicone. The new method is to manufacture the main body first and reserve a grid-like gap on the side of the body. The body is inverted into the mold and cast in one piece as a silicone film, ensuring an airtight seal. The reserved voids allow the silicone liquid to flow sufficiently to achieve the effect of homogenizing the size of the air cavity. In the exploration of the bistable structure, the size of the center circle of the sandwich disc, the number of surrounding teeth, the percentage of tooth area, and the staggered arrangement of upper and lower teeth are continuously adjusted to finally achieve a better bistable response.

1. 背景熟悉和前期工作

在暑期科研的初期阶段，我们对于课题并不了解并且未确定具体的研究方向。所以我们进行了较为宽泛的搜寻和潜在领域的方向探索。

1.1 背景熟悉

1.1.1 Kirigami 结构

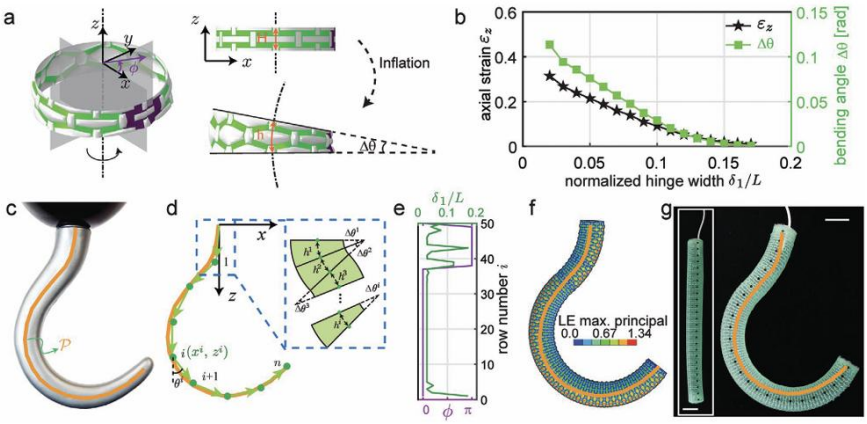


图 4-1 Kirigami

Kirigami 结构是源自日本的剪纸技术，通过改变纸张的结构可以实现许多有趣的动态活动。比如这篇文献中所展现的弯曲的钩子的结构，对于我们的研发颇有启发。

1.1.2 一些简单的造型

由于膨胀硅胶和普通硅胶附着在一起所产生的向特定方向弯曲的效果，促使我们去寻找类似的结构造型设计，可以引用我们的技术。主要的思路是该造型能在一个信号刺激下改变形状，并且改变的方向是可重复多次的、同一的。

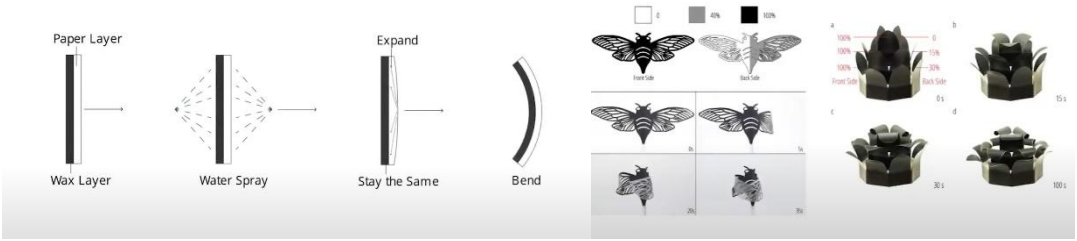


图 4-2

Wax paper 不同的灰度会造成不同的弯曲程度和弯曲速度。结合 rhino6 建模可以

模拟出不同 combination 和 shape 的弯曲。

视频中展示了很多 wax paper actuator 相关应用，包括电路、农业播种、装饰。利用这个材料在有水的情况下可弯曲，干燥情况下可恢复笔直的特性，可以输出一个 01 信号，从而应用到各个装置中。

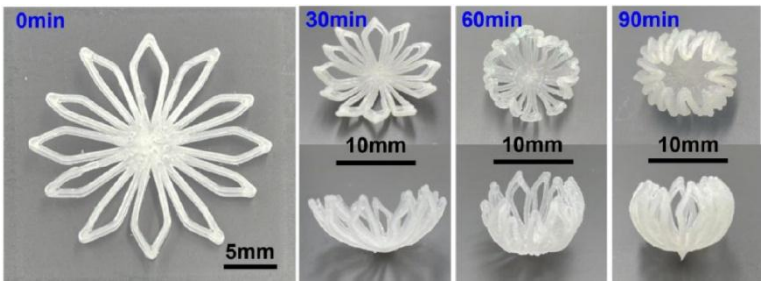


图 5-1

该造型是可开合的花朵，可用于制造装饰品，成本低且一体化制造，不易损坏。

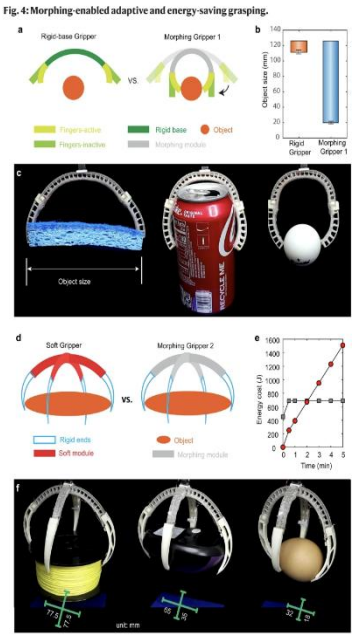


图 5-2

该文献描述了一种双层爪子可以轻松抓取物体，对于后续的硅胶臂的气腔设计有所启发。

1.2 前期工作

在了解了相关背景知识的基础上，我们开展了探究热膨胀硅胶形变的影响因素的相关工作，主要从膨胀硅胶的宽度（a）、长度（b）、厚度（h）、浓度（w%）四个

因素着手实验

变量参数	a（mm）	b（mm）	h（mm）	w%
a作为变量	4	40	2	8%
	6			
	8			
	10			
	12			
b作为变量	8	20	2	8%
		30		
		40		
		50		
		60		
h作为变量	8	40	1	8%
			1.5	
			2	
			2.5	
			3	
w%作为变量	8	40	2	4%
				8%
				12%
				16%
				20%

1.2.1 探究以上因素对硅胶形变程度的影响

实验方法：

A. 准备模具并且制作硅胶模型

以 h 为变量：

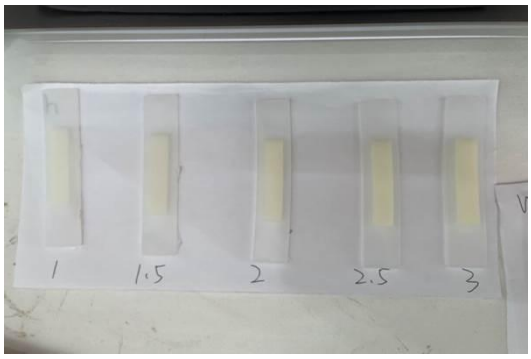


图 7-1

以 w%为变量：

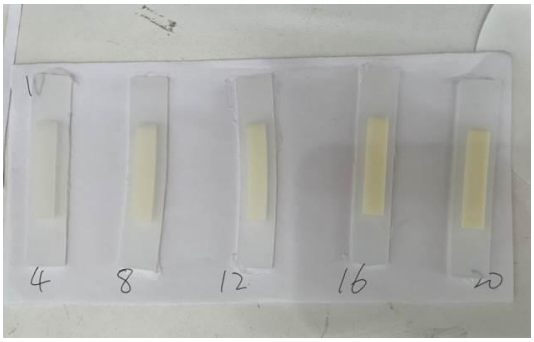


图 7-2

以 a 为变量：

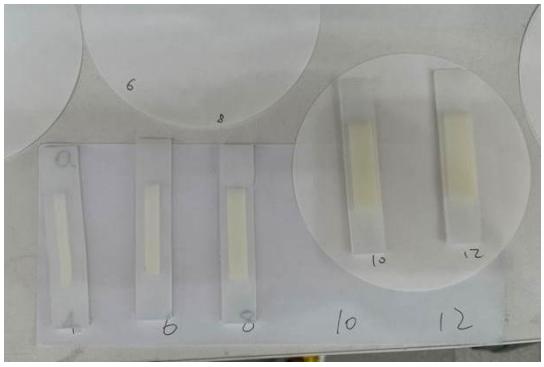


图 7-3

B. 加热膨胀硅胶

C. 测定曲率：通过测量形变后形成的近似圆弧的弯折曲率来反映形变程度（如图）

具体方法 1：测量拟合圆的半径 R（弯折曲率 1，见下表）

具体方法 2：测量膨胀硅胶长度/弯曲两点的直线距离，b/d（弯折曲率 2，不适用于以 b 为变量）

图 7-4 具体方法 1

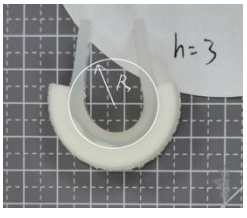


图 7-5 具体方法 2

具体方法 3：利用 Matlab 代码分析图像，并用差色图标注出各个位置的曲率，可用于粗略分析曲率。具体代码见附录。

原始图片：

图 8-1 具体方法 3

差色图：

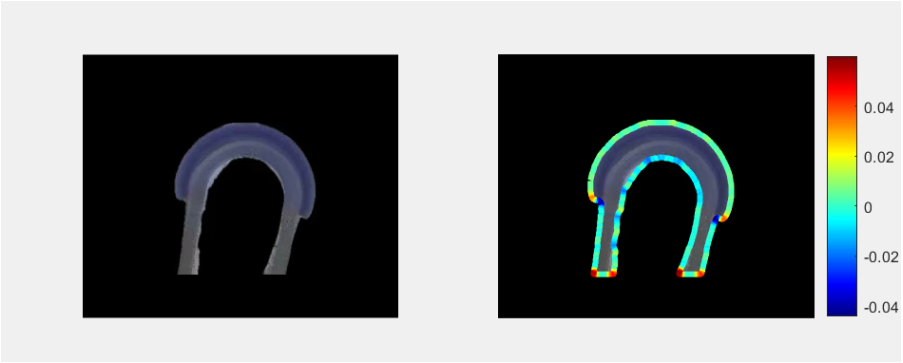


图 8-2 具体方法 3

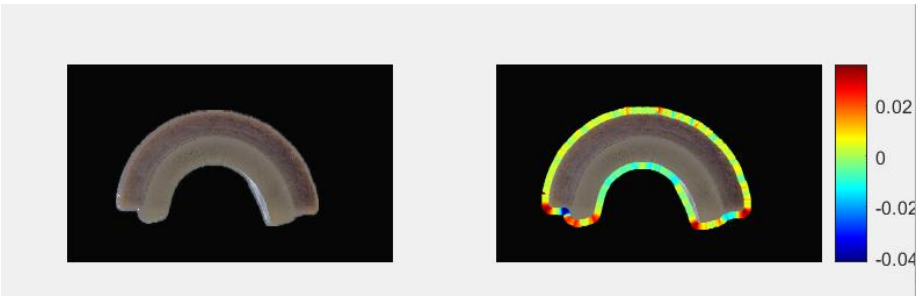


图 8-3 具体方法 3

该种方法可利用计算机精确显示每一个小部分的曲率，但是对于整体曲率的测算帮助不大。在经过改良后可以实现输入一张去掉背景的图片后，输出差色图。实验规模增大后可利用该方法显著减少工作量。但是在本次科研活动中，我们小组实验规模较小，更追求更科学的几何测算。

实验结果：

变量参数	a	b	h	w%	弯折曲率1（拟合圆的半径R）	弯折曲率2（新的比例1=膨胀硅胶的长度/弯曲两点的直线距离）（不适用于b为变量）
a作为变量	4	40	2	8%	1.21	1.9512
	6				1.09	2.6316
	8				1.14	2.2222
	10				1.01	2.6667
	12				0.89	4.2105
b作为变量	8	20	2	8%	1.42	
		30			1.32	
		40			1.13	
		50			1.28	
		60			1.51	
h作为变量	8	40	1	8%	1.36	1.4815
			1.5		1.20	1.9048
			2		1.02	3.0769
			2.5		1.18	1.9048
			3		1.34	1.5873
w%作为变量	8	40	2	4%	1.51	1.2903
				8%	1.16	2.1053
				12%	0.97	3.3333

				16%	1.00	2.8169
				20%	0.95	2.7586

数据处理：我们将各个变量对形变的影响以折线图的形式呈现变　化趋势。
*半径对应对应弯折曲率 1，比例对应弯折曲率 2

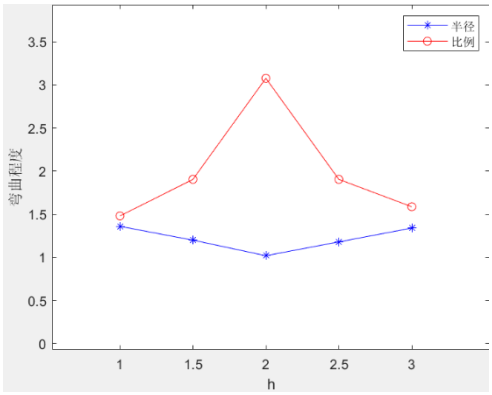


图 10-1 以厚度 h 为变量

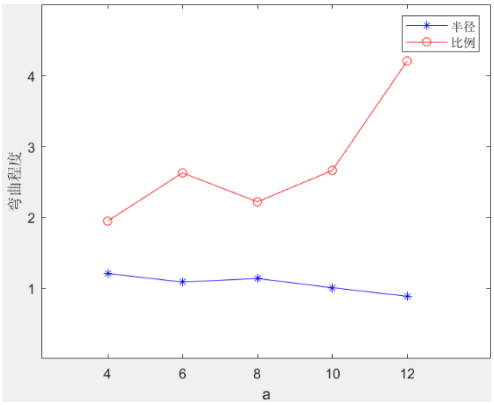


图 10-2 以宽度 a 为变量

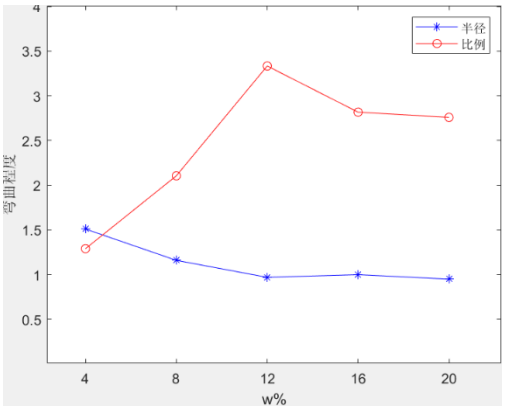


图 10-3 以浓度 w%为变量

总结：热膨胀硅胶弯曲程度与长度 b 关联性不强；与宽度 a、浓度 w%呈现正相关性；并且随着厚度 h 的增长，弯曲程度呈现先增后减的趋势。

1.2.2 探究以上因素对硅胶弯折应力的影响

实验方法：

A. 准备模具模型同上

B. 测量方法：通过测量将膨胀硅胶摊平后由于向上弯曲的趋势而形成的弹力大小来反映应力大小

具体方法：差力法（由于所测对象为软体，需要排除测量头和物体挤压所产生的误差）

a. 将待测模型如图固定在压敏计上，控制压敏计的示数 F1 在 0～0.1N 之间，在示数稳定时记录 F1

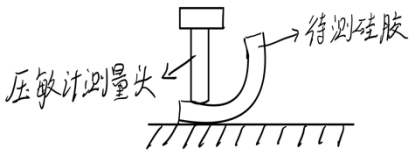


图 11-1

b. 将待测模型在另一端用手缓慢向下压，直至模型平整，待示数稳定时记录示数 F2

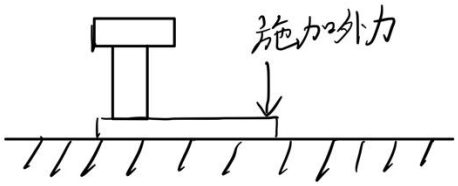


图 11-2

c. 弹力大小=F2-F1

实验结果

变量参数	a:mm	b:mm	h:mm	w%	弯曲应力（力传感器）：N
a 作为变量	4	40	2	8%	0.15
	6				0.21
	8				0.23
	10				0.29
	12				0.32

b 作为变量	8	20	2	8%	
		30			
		40			
		50			
		60			
h 作为变量	8	40	1	8%	0.15
			1.5		0.23
			2		0.15
			2.5		0.50
			3		0.43
w%作为变量	8	40	2	4%	0.09
				8%	0.35
				12%	0.32
				16%	0.39
				20%	0.50

总结：弯曲应力与宽度 a、厚度 h、浓度 w%呈现正相关性，与长度 b 关联性不大

2. 探究使用热致变形硅胶制作双稳态结构以及软体机器人硅胶臂

2.1 双稳态结构设计

2.1.1 背景介绍

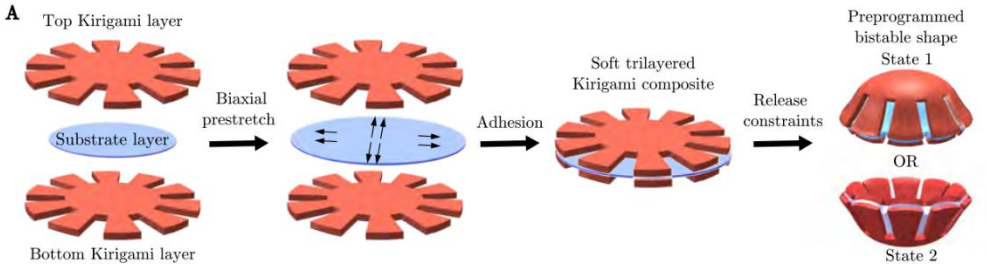


图 13-1 查阅论文得到的双稳态结构示意图[1]

基于论文 **Design of bistable soft deployable structures via a Kirigami-Inspired Planar fabrication approach** 中提到的双稳态结构设计，我们设计模具并制作了以下原型：

2.1.2 制作方法

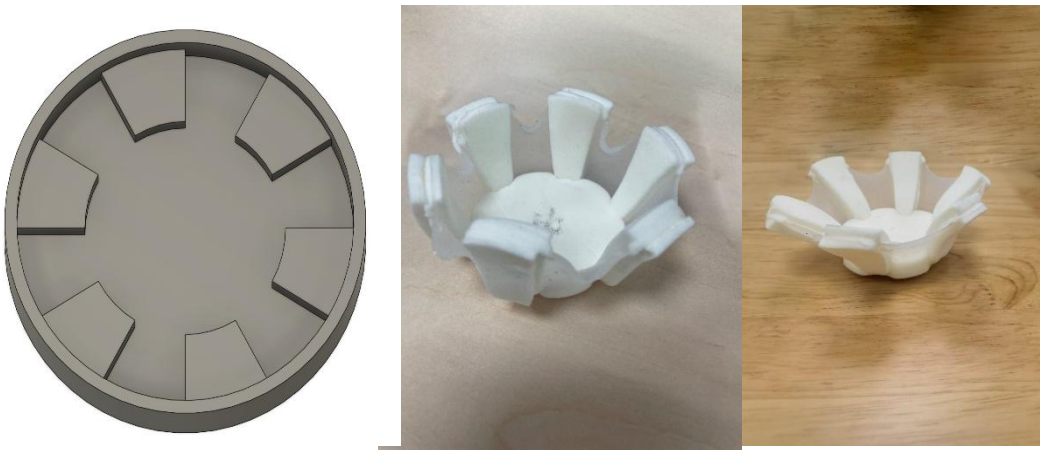


图 13-2 基于论文双稳态结构所制造的原型

制作方法为：向模具内填充一层指定浓度的膨胀硅胶作为变形层 1，待变形层 1 凝固后，填充一层普通硅胶作为支撑层。待支撑层凝固后，将模型脱出，向模具内填充一层不同指定浓度的膨胀硅胶作为变形层 2，再将模型倒扣入模具，使变形层 2 与支撑层相结合，从而完成整个原型的制作。

我们成功制造论文所呈现的双稳态结构，且能实现对应的功能：能实现两个稳态之间的相互转换。同时，我们也从原型获得启发，探索出了更多的研究方向。

双稳态 (变形层交错)	膨胀硅胶：厚度：3mm；浓度：12%；中...			
双稳态 (变形层交错)	膨胀硅胶：厚度：3mm；浓度：12%；中...			
双稳态 (改变齿数5)	膨胀硅胶：厚度：3mm；浓度：12%； ...			
双稳态 (改变齿数7)	膨胀硅胶：厚度：3mm；浓度：12%； ...			
双稳态 (两面夹持) 详情	膨胀硅胶：厚度：2mm；浓度：12%； ...			
双稳态 (两面夹持改良1)	膨胀硅胶：厚度：2mm；浓度：12%； ...			
双稳态 (两面夹持改良2)	膨胀硅胶：厚度：2mm；浓度：12%； ...			

对双稳态结构的进一步探究

我们大致从四个方面对双稳态结构进行了探究：改变变形层的排列方式，改变变形层中心圆大小，改变变形层齿数，改变变形方式。

我们尝试将变形层交错排列，发现其效果不如将变形层相对排列。交错的变形层反而不利于双稳态效果的实现。如下图所示：

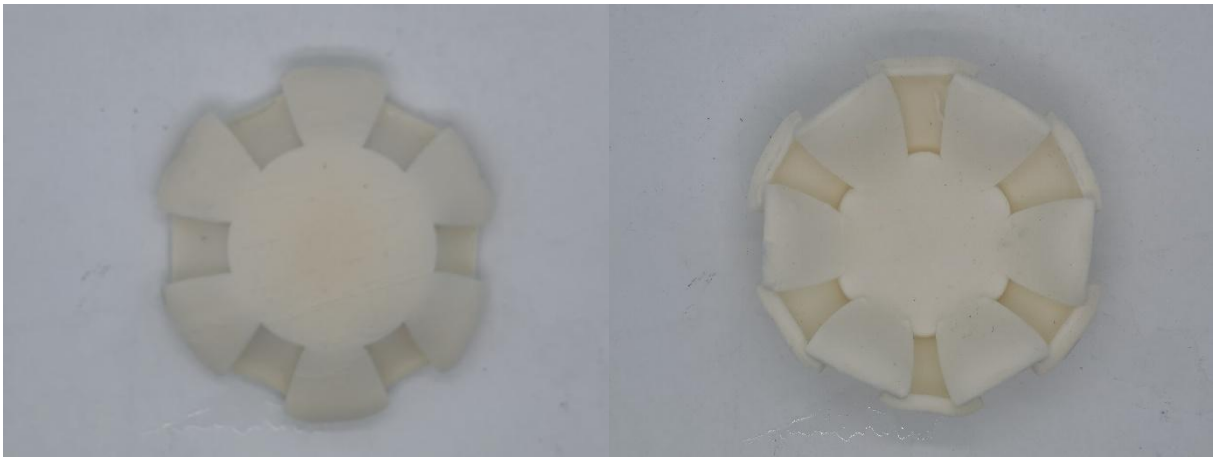


图 14-1 变形层交错对双稳态效果实现的影响

在此实验的基础上，我们进一步探究了中心圆大小的影响，我们设计了一系列变形层中心圆直径从 30mm 到 50mm 的双稳态结构，最终发现直径在 40mm 附近能较好实现双稳态效果。

我们尝试改变变形层的齿数，发现奇数齿数并不利于双稳态效果的实现。同时，低于 4 或是高于 8 的齿数也会导致双稳态效果无法实现。6 齿是一个较为合理的形状，能较好实现双稳态效果。如下图所示：

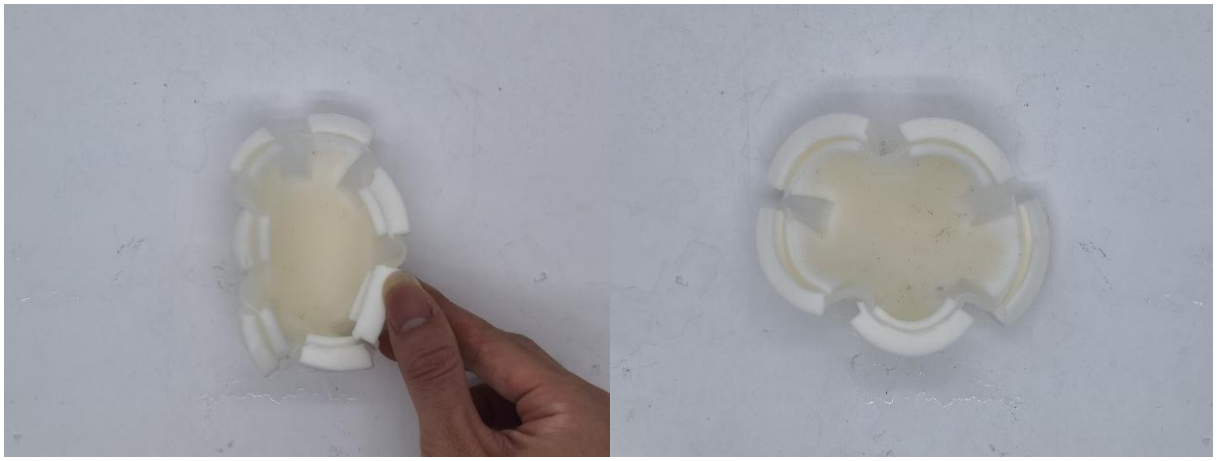


图 14-2 变形层齿数对双稳态效果实现的影响

我们尝试对双稳态结构的变形方式进行改变。具体实现方式为：在支撑层处添加用于夹持的小孔，模型制作完成后，利用两个圆环将支撑层夹持住，从而实现了整体变形到部分变形。如下图所示：

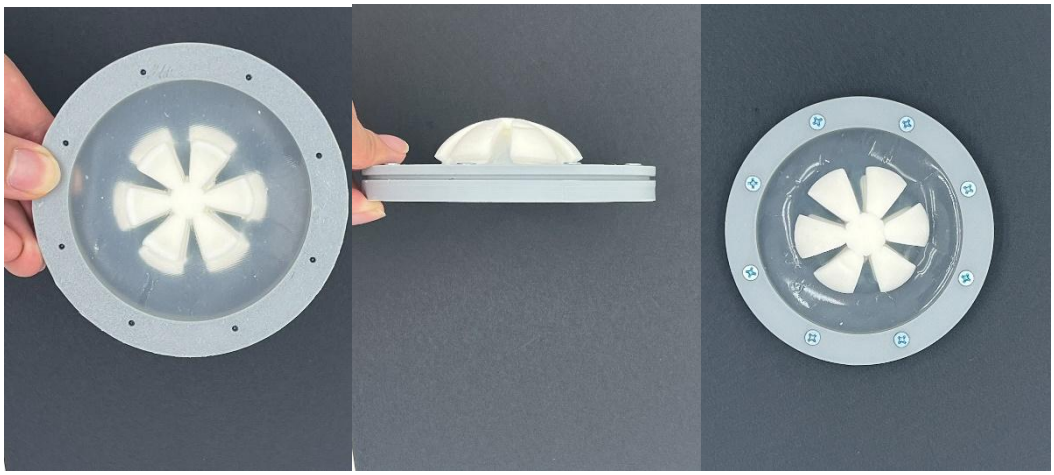


图 15-1

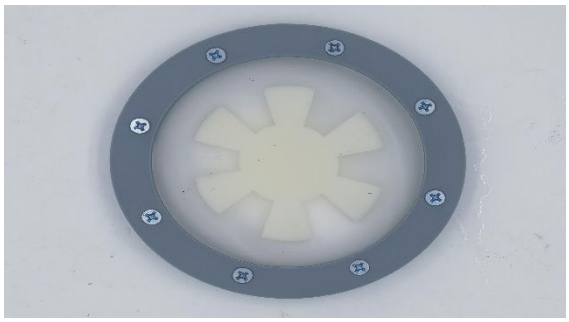


图 15-2

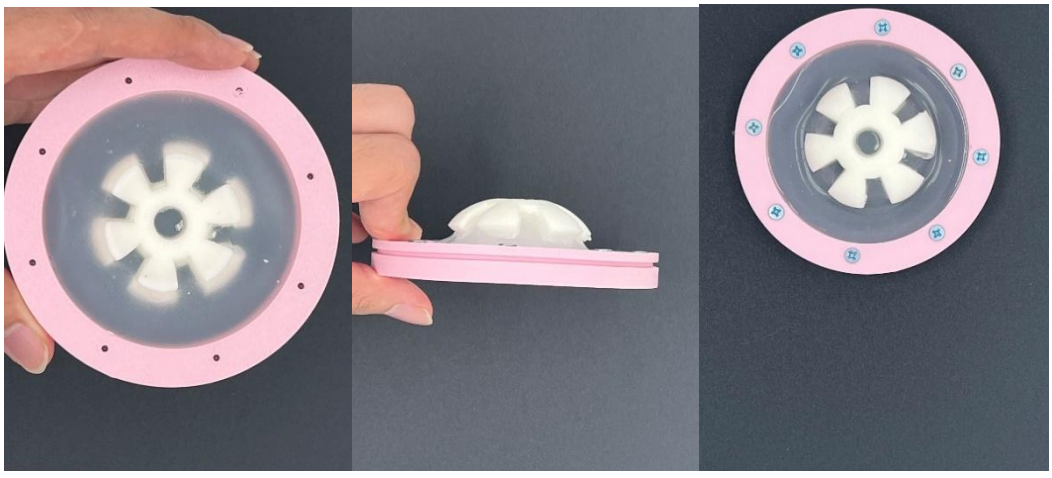
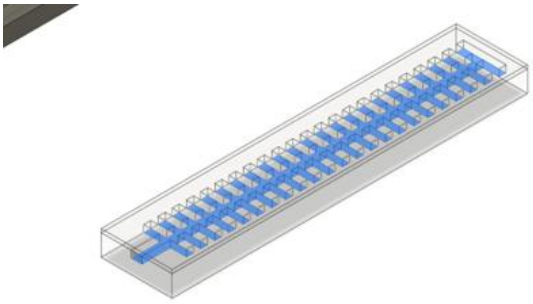


图 15-3　　变形方式对双稳态效果实现的影响

2.2 制作软体机器人硅胶臂

2.2.1 背景介绍

在软体机器人中，硅胶臂代替了传统机器人的机械臂，并依赖充气驱动。气动硅胶臂基本模型如图 16-1. 所示。



（图例：蓝色，气腔）

图 16-1 气动硅胶臂基本模型

基本弯曲原理是上下两面硅胶厚度不同，充气后的延展程度不同，形成向一边弯折的效果，如下图所示。

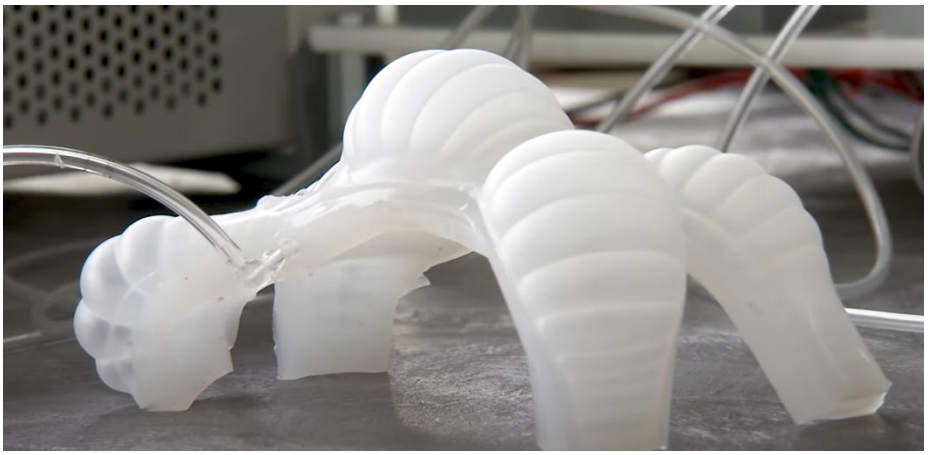


图 16-2.

结合先前实验，有另一种设计硅胶臂的思路。使用膨胀硅胶产生一个预形变，充气后反而转为平整。但是制作普通硅胶臂时，使用传统的两部分分别浇筑再粘连的方法，我们发现难以得到一个均一稳定的产品。其充气程度和弯曲程度对应关系不一致，这会导致后续难以结合电子控制系统组成完整的软体机器人，如下图所示。

. 图 17-1

所以我们设计了一个新的制作工艺，实现了均一稳定的基础版气动硅胶臂。

2.2.2 制作方法

1. 模具设计

以黄色表面为分界面，设计两个模具，如下图。

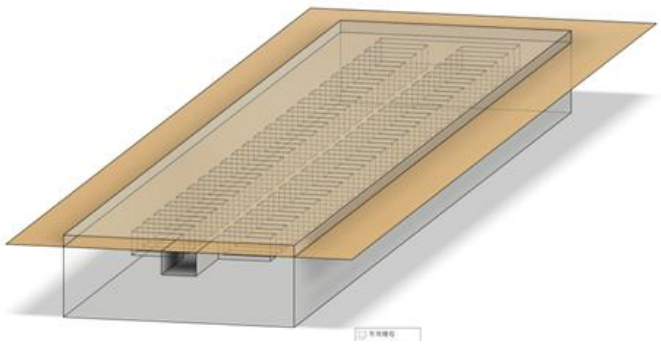


图 17-2 模具 1

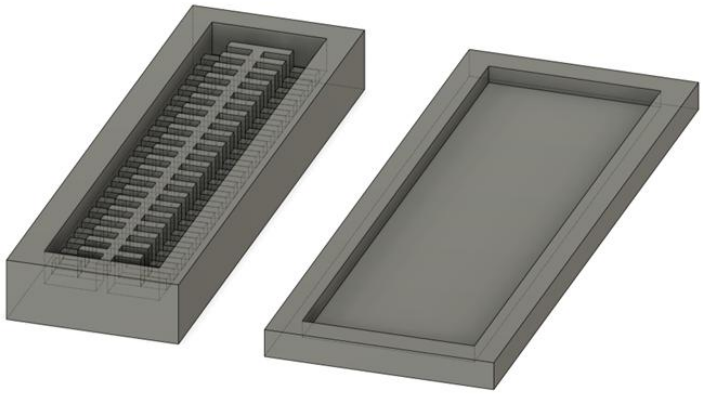


图 18-1　模具 2

2、 硅胶浇筑

a. 将硅胶倒入模具 1，去除气泡，使液面与模具边缘齐平。

b. 脱模后效果如下图所示。

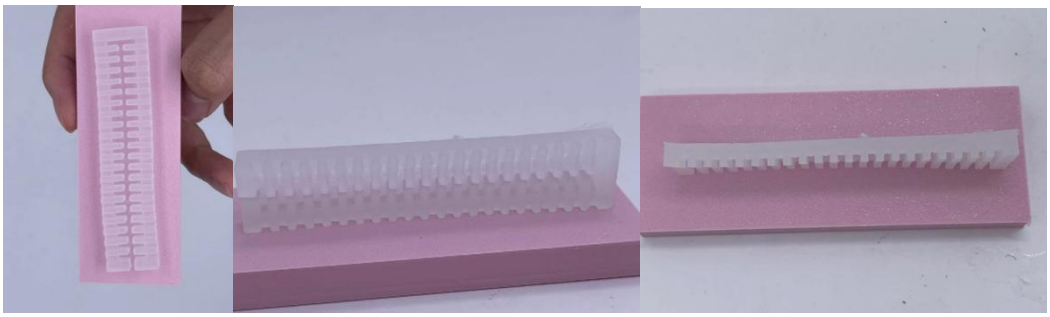


图 18-2 脱模后效果图

c. 将硅胶倒入模具 2，液面与边缘齐平，不等其凝固，就放入上一步的硅胶固体。放入硅胶时，注意先从一端慢慢放入，以便空气和硅胶能够充分流动。而且在放入时要注意不要压缩或膨胀半开的气腔(换句话说，在放入时尽量让气腔保持目标形状)，否则会导致模具中的液体硅胶被吸入气腔。

如下图所示

图 18-3

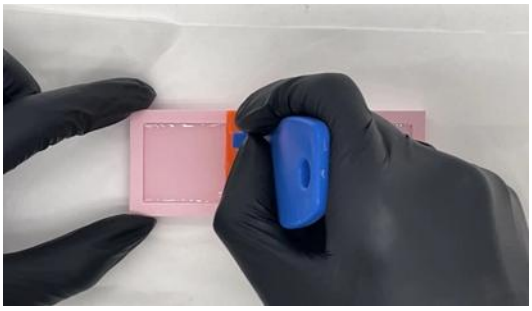


图 19-1

图 19-2

d.脱模后剪去多余的部分，如下图所示。

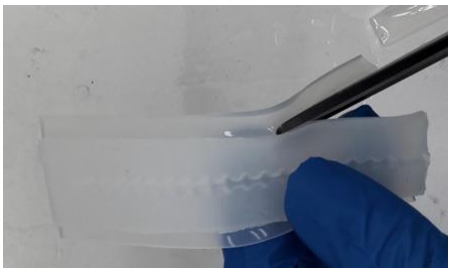


图 19-3　剪去多余部分

3．最终成果和结论

设计模具是核心步骤，总体思路是在气腔的上下表面划分基底。控制模具的深度刚好能使气腔结构在浸入硅酮固体后高于液面。局限性：模具数量较多，无法处理气腔的不规则分布。

图 19-4 成品

结论

在广泛阅读多方向文献和动手实践后，我们确定了软体机器人和双稳态结构的两个研究方向。在此之前我们对于膨胀硅胶产生的弯曲作用进行了一系列基础数据的测定，如膨胀硅胶的浓度和大小对于弯曲程度和弯曲应力的影响。这些数据可以尝试归纳出一个经验公式，方便后续具体结构的设计。对于气动硅胶臂的制造问题，具体表现为气腔大小不均一、气密性差，我们优化了制作工艺，解决了该问题。在双稳态结构研究中，我们以夹层圆盘为研究对象，通过不断改变参数实现了两种稳态之间切换快速、稳定。该结构可类比输出或接收 01 信号，应用于多种场景中。

引用文献

图4-1：Jin, L., Forte, A. E., Deng, B., Rafsanjani, A., & Bertoldi, K. (2020). Kirigami-Inspired Inflatables with Programmable Shapes. *Advanced Materials*, *32*(33). https://doi.org/10.1002/adma.202001863

图4-2：ACM SIGCHI. (2024, May 9). *Waxpaper Actuator: sequentially and conditionally programmable wax paper for morphing interfaces* [Video]. YouTube. https://www.youtube.com/watch?v=IoOU0Sz07Lc

图5-1：Zhang, C., Liao, E., Li, C., Zhang, Y., Chen, Y., Lu, A., Liu, Y., & Geng, C. (2023). 3D Printed Silicones with Shape Morphing and Low-Temperature Ultraelasticity. *ACS Applied Materials & Interfaces*, *15*(3), 4549－4558. https://doi.org/10.1021/acsami.2c20392

图5-2：Sun, J., Lerner, E., Tighe, B., Middlemist, C., & Zhao, J. (2023). Embedded shape morphing for morphologically adaptive robots. *Nature Communications*, *14*(1). https://doi.org/10.1038/s41467-023-41708-6

图13-1：*Mungekar, M., Ma, L., Yan, W., Kackar, V., Shokrzadeh, S., & Jawed, M. K. (2023). Design of bistable soft deployable structures via a Kirigami－Inspired Planar fabrication approach. Advanced Materials Technologies, 8(16). https://doi.org/10.1002/admt.202300088*

图17-1：NASA Langley Research Center. (2019, September 17). *Life at the lab: soft robots* [Video]. YouTube. https://www.youtube.com/watch?v=iwQRYzLZvGE

Matlab code: Driscoll MK, McCann C, Kopace R, Homan T, Fourkas JT, Parent C, et al. (2012) Cell Shape Dynamics: From Waves to Migration. PLoS Comput Biol 8(3): e1002392. https://doi.org/10.1371/journal.pcbi.1002392

附录

文件1：curve-fit.m

```
    clear; close all; clc;
    Start = tic;

    %% Inputs
    %-------------------------------------------------------------------------
--
    % Image files
    I = imread('h=1.jpg');    %'2019_04_19_19_33_26_456_Ang_0.bmp');
    I_reverse = imcomplement(I);

    % Measure shape parameters
    boundaryPoint = 10;          %12, 10 number of boundary points curvature is
found over
    curvatureThresh  =  0.25;         %0.06  the  maximum  allowed  value  of  the
curvature measure
    bp_tangent = 10;             % number of boundary points the tangent angle
is found over
    interpdmin = 0.3 ;             % the minimum number of pixels seperating
boundary points after interpolation
    loopclose = 0;                 % 0 - if open boundaries | 1 - if closed
boundaries



    %% Find the curvature
    [shape_details,    Icurv]    =    curvature(I_reverse,    boundaryPoint,
curvatureThresh, bp_tangent, ...
```

```
                                    interpdmin, loopclose);

    %% Plot curvature
    X = shape_details.XY(:,1);
    Y = shape_details.XY(:,2);
    Z = zeros(size(X));
    C = shape_details.curvature'*1;

    % Plot
    figure;
    s1 = subplot(1,2,1); imshow(Icurv)
    s2 = subplot(1,2,2);
    imshow(Icurv)
    hold on
    surf([X(:) X(:)], [Y(:) Y(:)], [Z(:) Z(:)], [C C], ...   % Reshape and
replicate data
     'FaceColor', 'none', ...    % Don't bother filling faces with color
     'EdgeColor', 'interp', ...  % Use interpolated color for edges
     'LineWidth', 3);            % Make a thicker line
    hold off

    cmap = jet;
    colormap(cmap);
    cb = colorbar;  % Add a colorbar
    cb.Label.String = 'Curvature';

    s1Pos = get(s1,'position');
    s2Pos = get(s2,'position');
    s2Pos(3:4) = [s1Pos(3:4)];
    set(s2,'position',s2Pos);
```

```
x1=487;
x2=737;
y1=549;
y2=253;
[index_x1]= find(abs(X-x1)<0.001);
[index_x2]= find(abs(X-x2)<0.001);
[index_y1]= find(abs(Y-y1)<0.001);
[index_y2]= find(abs(Y-y2)<0.001);



%plot X and Y with their index
% x=1:1:953;
% a=X;
% b=Y;
% plot(x,a,'-*b',x,b,'-or');%线性，颜色，标记
% axis([0,960,0,900])  %确定x轴与y轴框图大小
% set(gca,'XTick',[0:100:1000]) %x轴范围1-6，间隔1
% set(gca,'YTick',[0:100:900]) %y轴范围0-700，间隔100
% legend('X','Y');   %右上角标注
% xlabel('索引号')  %x轴坐标描述
% ylabel('变量值') %y轴坐标描述

%% End parameters
%--------------------------------------------------------------------------
--
Runtime = toc(Start);
```

文件2：curvature.m

```
    function [shape, Icurv] = curvature(image, boundaryPoint, curvatureThresh, ...
                                        bp_tangent, interp_resolution, loopclose)

    %%*************************************************************************%
    %*                          Curvature measure                              *%
    %*                  Measure shape properties of loops.                     *%
    %*                                                                         *%
    %* Original author: Dr. Meghan Driscoll                                    *%
    %* Modified by: Preetham Manjunatha                                        *%
    %* Github link: https://github.com/preethamam
    %* Date: 08/02/2021                                                        *%
    %*************************************************************************%
    %
    %*************************************************************************%
    %
    %   Usage: [shape, Icurv] = curvature(image, boundaryPoint, curvatureThresh, ...
    %                                     bp_tangent, interp_resolution, loopclose)
    % Inputs: % Inputs:
    %           Image               - input image
    %            boundaryPoint        - number of boundary points over which curvature is found
    %           curvatureThresh     - the largest curvature magnitude allowed in the cutoff curvature
```

```
    %             bp_tangent            - the number of boundary points over
which the boundary tangent
    %                                     direction is measured
    %           interp_resolution     - interpolation resolution -- the minimum
number of pixels seperating
    %                                     boundary points after interpolation
    %           loopclose             - 0 - if open boundaries | 1 - if closed
boundaries
    %
    % Outputs:
    %           shape
    %           .curvature          - the boundary curvature at each boundary
point (uses snakeNum)
    %                                     Curvatures above or below a cutoff are
given the magnitude of the cutoff
    %           .meanNegCurvature   - the mean negative curvature
    %             .numIndents           - the number of boundary regions over
which the curvature is negative
    %           .tangentAngle       - the angle of the tangent to the boundary
at each boundary point
    %           .tortuosity         - the boundary tortuousity (a measure of
how bendy the boundary is)
    %             .uncutCurvature     - the uncut boundary curvature at each
boundary point (uses snakeNum)

    %             Icurv                 - Output image (padded image to make 3
channel. This fixes the plot)
    %-------------------------------------------------------------------------
--
    %
```

```
%%%%%%%%%%%%%%%% MEASURE SHAPE %%%%%%%%%%%%%%%%
% Original author: Dr. Meghan Driscoll
% Modified and compacted/concised a complicated codebase by: Preetham Manjunatha
%
% Thanks to Dr. Meghan Driscoll who kindly shared her code for academic purpose.
% If you use this code for visualization and other academic/research/any purposes.
%
% Please cite:
%
% Reference:
% Driscoll MK, McCann C, Kopace R, Homan T, Fourkas JT, Parent C, et al. (2012)
% Cell Shape Dynamics: From Waves to Migration.
% PLoS Comput Biol 8(3): e1002392.
% https://doi.org/10.1371/journal.pcbi.1002392
%
% Important note: This code uses parfor to speed up the things. If you do not have
% the Matlab parallel computing toolbox. Please make 'parfor' as 'for' in this
% function.
%
% %%%%%%%% This code is way too slow! (curvature should not be in a for loop) %%%%%%%%%
% If I have time I will try to improve this. If anyone improves it, please
% share the modified version code with me.
```

```
% RGB to binarization
if(size(image,3) == 3)
    Igray = rgb2gray(image);
    binaryimage = imbinarize(Igray);
    Icurv = image;
elseif (islogical(image))
    binaryimage = image;
    Icurv = im2uint8(repmat(image,1,1,3));
else
    binaryimage = imbinarize(I);
    Icurv = im2uint8(repmat(image,1,1,3));
end

% Find boundaries X and Y coordinates
cc_index = 1;
boundaries = bwboundaries(binaryimage,8);
x = boundaries{cc_index}(:, 2);
y = boundaries{cc_index}(:, 1);

% Perimeter of the binary component
stats = regionprops(binaryimage,'perimeter');
perimeter = cat(1,stats(cc_index).Perimeter);

% Interpolate for more points
% Every coordinates
xn = x+rand(size(x))*1e-8;
yn = y+rand(size(x))*1e-8;
[xi,yi] = snakeinterp1(xn,yn,interp_resolution);

% Every nth coordinates
```

```
    switch loopclose
        case 0
            xn = xi(1:boundaryPoint:end);
            yn = yi(1:boundaryPoint:end);
        case 1
            xn = [xi(1:boundaryPoint:end), xi(1)];
            yn = [yi(1:boundaryPoint:end), yi(1)];
    end
    shape_XY = [xn;yn]';
    M = numel(xn);

    % initialize variables
    shape_curvature         = NaN(1,M);
    shape_uncutCurvature    = NaN(1,M);
    shape_meanNegCurvature  = NaN(1,1);
    shape_numIndents = NaN(1,1);
    shape_tortuosity = NaN(1,1);
    shape_tangentAngle = NaN(1,M);

    % calculate the curvature (by finding the radius of the osculating circle using three neaby boundary points)
    bp_positions  =  [shape_XY(M-1-boundaryPoint:M-1,:);  shape_XY(1:M-1,:); shape_XY(1:boundaryPoint+1,:)];
    parfor j = 1:M

        % assign the three points that the circle will be fit to such that the slopes are not infinite
        point1 = bp_positions(j,:);
        point2 = bp_positions(j+boundaryPoint,:);
        point3 = bp_positions(j+2*boundaryPoint,:);
```

```
        slope12 = (point1(1,2)-point2(1,2))/(point1(1,1)-point2(1,1));
        slope23 = (point2(1,2)-point3(1,2))/(point2(1,1)-point3(1,1));

        if slope12==Inf || slope12==-Inf || slope12 == 0
            point0 = point2; point2 = point3; point3 = point0;
            slope12 = (point1(1,2)-point2(1,2))/(point1(1,1)-point2(1,1));
            slope23 = (point2(1,2)-point3(1,2))/(point2(1,1)-point3(1,1));
        end

        if slope23==Inf || slope23==-Inf
            point0 = point1; point1 = point2; point2 = point0;
            slope12 = (point1(1,2)-point2(1,2))/(point1(1,1)-point2(1,1));
            slope23 = (point2(1,2)-point3(1,2))/(point2(1,1)-point3(1,1));
        end

        % if the boundary is flat
        if slope12==slope23
            shape_curvature(1,j) = 0;

        % if the boundary is curved
        else

            % calculate the curvature
            x_center              =              (slope12*slope23*(point1(1,2)-
point3(1,2))+slope23*(point1(1,1)+point2(1,1))...
                        -slope12*(point2(1,1)+point3(1,1)))/(2*(slope23-
slope12));
            midpoint12 = (point1+point2)/2;
            midpoint13 = (point1+point3)/2;
```

```
            y_center = (-1/slope12)*(x_center-
midpoint12(1,1))+midpoint12(1,2);
            shape_uncutCurvature(1,j) = 1/sqrt((point1(1,1)-
x_center)^2+(point1(1,2)-y_center)^2);

            % cutoff the curvature (for visualization)
            shape_curvature(1,j) = shape_uncutCurvature(1,j);
            if shape_curvature(1,j) > curvatureThresh
                shape_curvature(1,j) = curvatureThresh;
            end

            % determine if the curvature is positive or negative
            [In, On] =
inpolygon(midpoint13(1,1),midpoint13(1,2),shape_XY(:,1),shape_XY(:,2));

            if ~In
                shape_curvature(1,j) = -1*shape_curvature(1,j);
                shape_uncutCurvature(1,j) = -1*shape_uncutCurvature(1,j);
            end

            if On || ~isfinite(shape_uncutCurvature(1,j))
                shape_curvature(1,j) = 0;
                shape_uncutCurvature(1,j) = 0;
            end

        end

    end
```

```
    % find the mean negative curvature (really this should use a constant dist
snake)
    listCurve = shape_uncutCurvature(1,1:M-1);
    listNegCurve = abs(listCurve(listCurve < 0));
    if ~isempty(listNegCurve)
        shape_meanNegCurvature(1,1) = sum(listNegCurve)/(M-1);
    else
        shape_meanNegCurvature(1,1) = 0;
    end

    % find the number of negative boundary curvature regions
    curveMask = (listCurve < 0);
    curveMaskLabeled = bwlabel(curveMask);
    numIndents = max(curveMaskLabeled);
    if curveMask(1) && curveMask(end)
        numIndents  = numIndents-1;
    end
    shape_numIndents(1,1) = numIndents;

    % find the tortuosity (should correct units)
    shape_tortuosity(1,1)      =      sum(gradient(shape_uncutCurvature(1,1:M-
1)).^2)/perimeter;

    % calculate the direction of the tangent to the boundary
    bp_positions_tangent=[shape_XY(M-1-bp_tangent:M-1,:);  shape_XY(1:M-1,:);
shape_XY(1:bp_tangent+1,:)];
    for j=1:M
        point1 = bp_positions_tangent(j,:);
        point2 = bp_positions_tangent(j+2*bp_tangent,:);
```

```
        shape_tangentAngle(1,j)      =      mod(atan2(point1(1,2)-point2(1,2),
point1(1,1)-point2(1,1)), pi);
    end

    shape.XY = shape_XY;
    shape.curvature          = shape_curvature;
    shape.uncutCurvature     = shape_uncutCurvature;
    shape.meanNegCurvature   = shape_meanNegCurvature;
    shape.numIndents = shape_numIndents;
    shape.tortuosity = shape_tortuosity;
    shape.tangentAngle = shape_tangentAngle;

    end



    %% Auxillary fucntions
    %-------------------------------------------------------------------------
--
    function [xi,yi] = snakeinterp1(x,y,RES)
    % SNAKEINTERP1  Interpolate the snake to have equal distance RES
    %     [xi,yi] = snakeinterp(x,y,RES)
    %
    %     RES: resolution desired

    %     update on snakeinterp after finding a bug

    %      Chenyang Xu and Jerry L. Prince, 3/8/96, 6/17/97
    %      Copyright (c) 1996-97 by Chenyang Xu and Jerry L. Prince
    %      image Analysis and Communications Lab, Johns Hopkins University
```

```
    % convert to column vector
    x = x(:); y = y(:);

    N = length(x);

    % make it a circular list since we are dealing with closed contour
    x = [x;x(1)];
    y = [y;y(1)];

    dx = x([2:N+1])- x(1:N);
    dy = y([2:N+1])- y(1:N);
    d = sqrt(dx.*dx+dy.*dy);  % compute the distance from previous node for
point 2:N+1

    d = [0;d];   % point 1 to point 1 is 0

    % now compute the arc length of all the points to point 1
    % we use matrix multiply to achieve summing
    M = length(d);
    d = (d'*uppertri(M,M))';

    % now ready to reparametrize the closed curve in terms of arc length
    maxd = d(M);

    if (maxd/RES<3)
       error('RES too big compare to the length of original curve');
    end

    di = 0:RES:maxd;
```

```
xi = interp1(d,x,di);
yi = interp1(d,y,di);

N = length(xi);

if (maxd - di(length(di)) <RES/2)  % deal with end boundary condition
   xi = xi(1:N-1);
   yi = yi(1:N-1);
end
end


function q = uppertri(M,N)                          %added by Ilya
% UPPERTRI   Upper triagonal matrix
%            UPPER(M,N) is a M-by-N triagonal matrix

[J,I] = meshgrid(1:M,1:N);
q = (J>=I);
end
```