รายงาน

Computer Assignment 3

จัดทำโดย

นาย ธนาคม หัสแดง รหัสนักศึกษา 590610624

เสนอ

ผศ.ดร. ศันสนัย์ เอื้อพันธ์วิริยะกุล

รายงานฉบับนี้เป็นส่วนหนึ่งของรายวิชา

Digital Image Processing (261453)

ภาควิชาคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

มหาวิทยาลัย เชียงใหม่

Introduction

รายงานเล่มนี้เป็นส่วนหนึ่งของรายวิชา 261453 Digital Image Processing โดยเป็นกำหนดให้สามารถ ใช้เทคนิคหรือวิธีตามต้องการไม่ว่าจะมาจากบทเรียนในห้องหรือค้นคว้าเอง เพื่อหาจำนวน และตำแหน่งของรูหนอนบนมะเขือม่วงในรูปที่กำหนด(ภาคผนวกรูปภาพ ก,ข) โดยในการจัดทำรางานนี้เป็นการค้นคว้าด้วยตัวเองและไม่ได้ใช้ Toolbox หรือ Program ใด ๆ ช่วยในการค้นหาแม้แต่น้อย อาจมีข้อผิดพลาดในบางผลการทดลอง ต้องขออภัยมา ณ ที่นี้ด้วย

ผู้จัดทำ

นายธนาคม หัสแดง

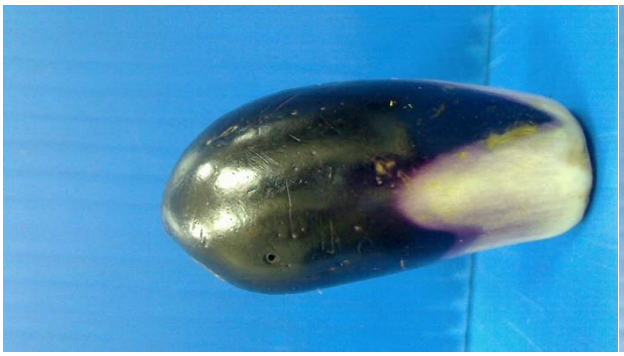
590610624

สารบัญ

Method

วิธีการที่ใช้ในการทดลองนี้ มี ด้วยกัน 3 ส่วนโดยจะอยู่ใน ลูปของ การคำนวณเดียวกัน

เริ่มต้น จะรับรูปที่กำหนดเข้ามาในโปรแกรม



WormHole_1H

WormHole_2H

ขั้นตอนต่อไป นำรูปที่ได้มาหมุน เนื่องจากเป็นรูปแนวนอน ซึ่งอาจไม่ต้องทำขั้นตอนนี้ก็ได้ แต่เพื่อให้สามารถมองรูปได้ชัดเจนจึงทำการหมุนให้กลับมาอยู่ในแนวตั้ง

WormHole_1H

WormHole_2H

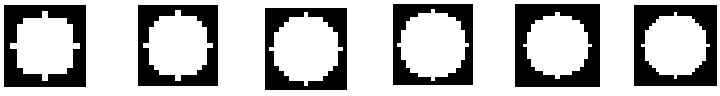
หลังจากนั้นแปลงรูปจาก RGB ให้เป็น BW และทำ Binary รูป โดยใช้ค่าสัมประสิทธิ์เป็น 0.23 แล้วกลับ BW ค่าเพื่อให้ง่ายต่อการคำนวณ

WormHole_1H

WormHole_2H

จากนั้นจะเริ่มเข้า ลูป โดยจะกำหนดตัวแปล(Circle_size) ในการวนลูปเป็น ขนาดของรูปวงกลมที่จะนำมาคำนวณ

ขั้นแรก ในการเข้าลูปเราจะสร้าง Array ขนาดเท่ากับ Circle_size โดยจะกำหนดค่าใน Array ให้ออกมาเป็นรูป วงกลม เพื่อใช้เป็น Kernel ในการหาวงกลมในรูปมะเขือม่วง โดยในที่นี้กำหนดขนาด Array เป็น 13 15 17 19 21 23 ดังรูป



13 15 17 19 21 23

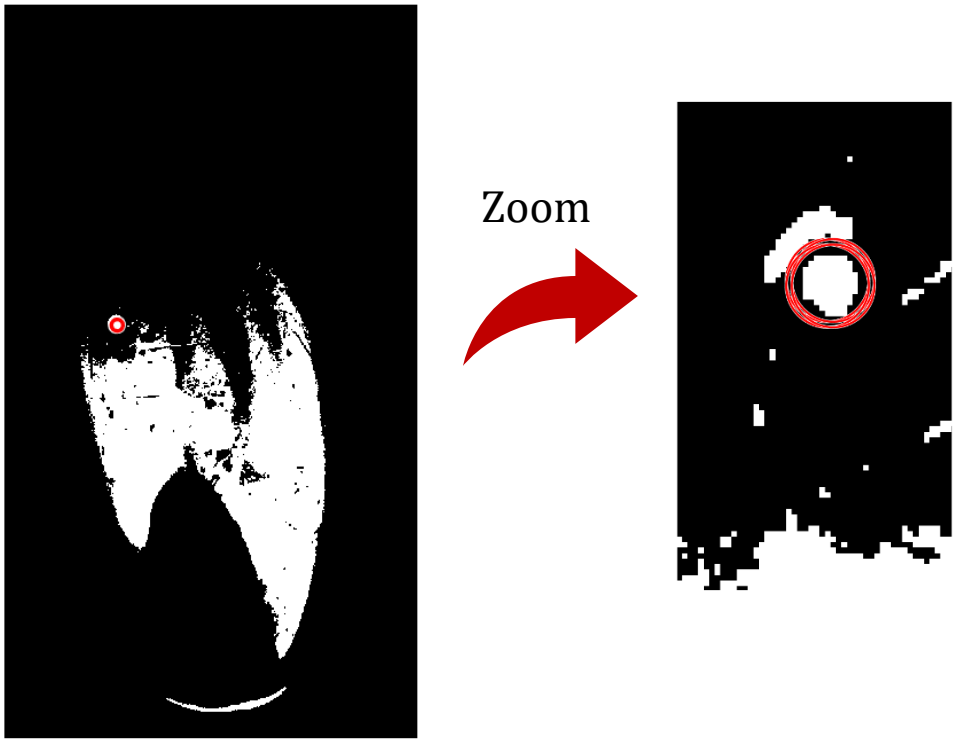
ขั้นสอง นำรูปมะเขือม่วงมา Pad ด้วย ขนาดของ Circle_size/2 ทั้ง 2 ข้างเพื่อจะทำ Convolution

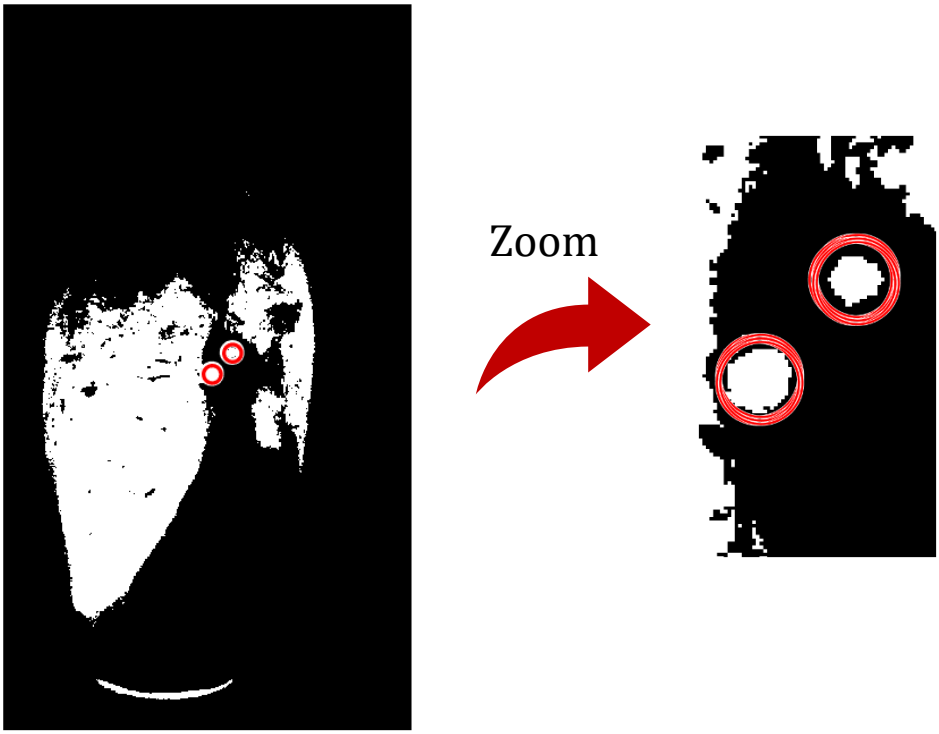
ขั้นสาม นำ Kernel ของวงกลม มาเข้าลูปคำนวณความเหมือนกันของ Kernel และรูปมะเขือม่วง กำหนดเงื่อนไขว่าถ้า Pixel นั้น ๆมีค่าตรงกันให้เก็บค่า avg เพิ่มขึ้นทีละ 1

ขั้นสี่ เมื่อจบแต่ละ Pixel จะนำเอาค่า avg มาเช็คกับค่ามากสุดที่จะเหมือนได้ระหว่าง Kernel และ รูปมะเขือม่วง โดยจะเทียบเพียง ระหว่าง 0.87 – 0.95 เท่านั้น เพราะถ้าน้อยกว่านี้อาจจะได้ตำแหน่งที่ไม่ใช่รูหนอนก็ได้ โดยเมื่อเทียบแล้วค่าได้ตรงตามเกณ ก็จะนำตำแหน่งของจุดนั้นไปเก็บไว้เพื่อบอกว่าเป็นจุดที่พบรูหนอน

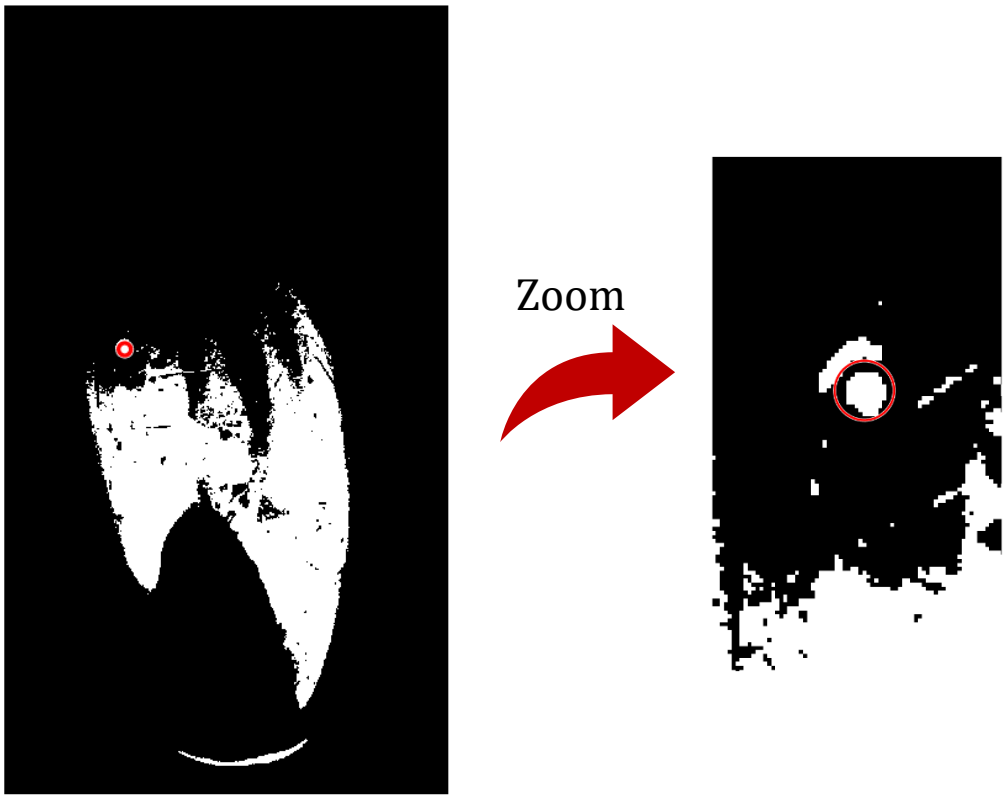
ซึ่งจะวนลูปนี้ไปจนครบจำนวนขนาดวงกลมที่ต้องการเช็ค

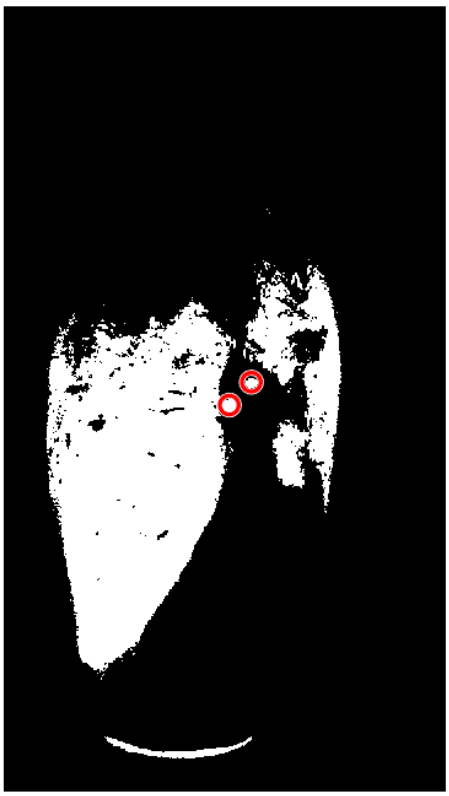
ขั้นต่อไป จะนำตำแหน่งของรูปหนอนที่ได้มาวาดลงไปบนรูป แต่จะเห็นว่า มีตำแหน่งที่ใกล้เคียงกันจนแทบจะเป็นจุดเดียวกัน อยู่หลายอันมาก ดังรูป





ดันนั้น จึงต้องแก้ปัญหาโดย ให้เช็คค่าว่า ปัจจุบันมี วงกลมอยู่หรือไม่ ถ้าไม่ให้วาดได้เลย แต่ถ้ามีอยู่ให้เช็คกับตัวปัจจุบันว่า ตำแหน่งอยู่หางกันเกิน ขนาดของ วงกลมหรือไม่ ถ้าเกินก็ให้วาดได้ แต่ถ้าไม่เกินจะข้ามวงกลมนั้นไปเลย เพื่อกันส่วนซับซ้อนของวงกลมที่อาจจะเกิดขึ้นได้ดังรูป





Zoom

Result

เมื่อนำรูปInput ที่หมุนแล้วมาเทียบ จะเห็นว่าได้จำนวนของรูหนอนและตำแหน่งของรูหนอนตามต้องการ ดังรูป โดยตำแหน่งของรูหนอนจะถูกวงกลมด้วยวงกลมสีแดง

WormHole_1H

WormHole_2H

และตำแหน่งของรูหนอนจะอยู่ที่ พิกัด ของรูปภาพ คือ

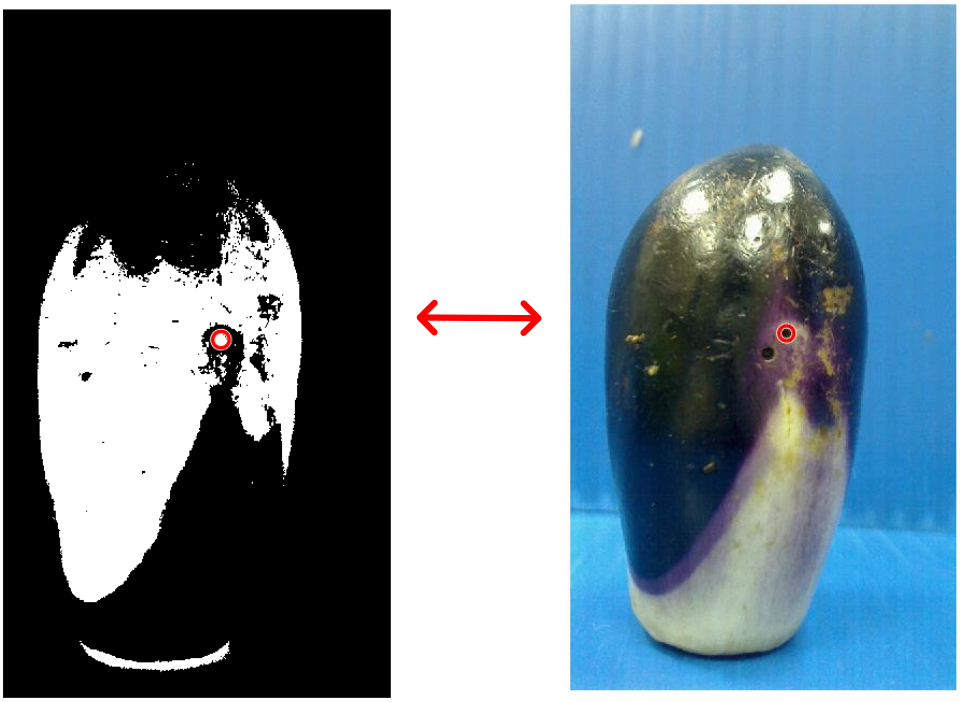
WormHole_1H : [147 , 419]

WormHole_2H : [303 , 460 ; 276 , 488]

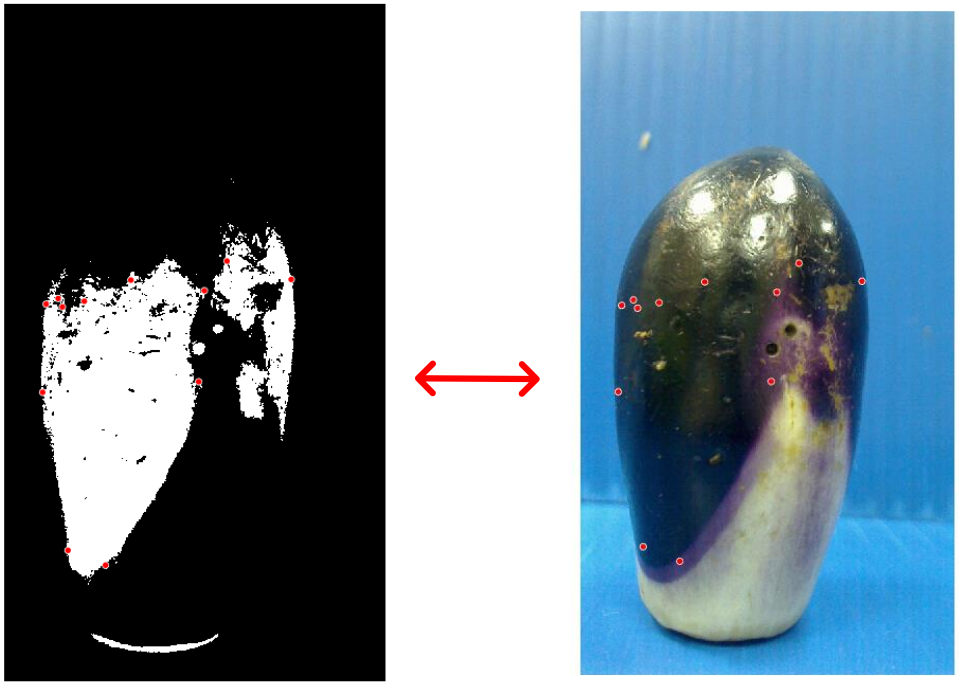
Discussion

จากการค้นคว้าโค้ดที่ได้ทำไป จะมีข้อผิดพลาดอยู่หลายประการ ขึ้นอยู่กับหลายปัจจัย คือ

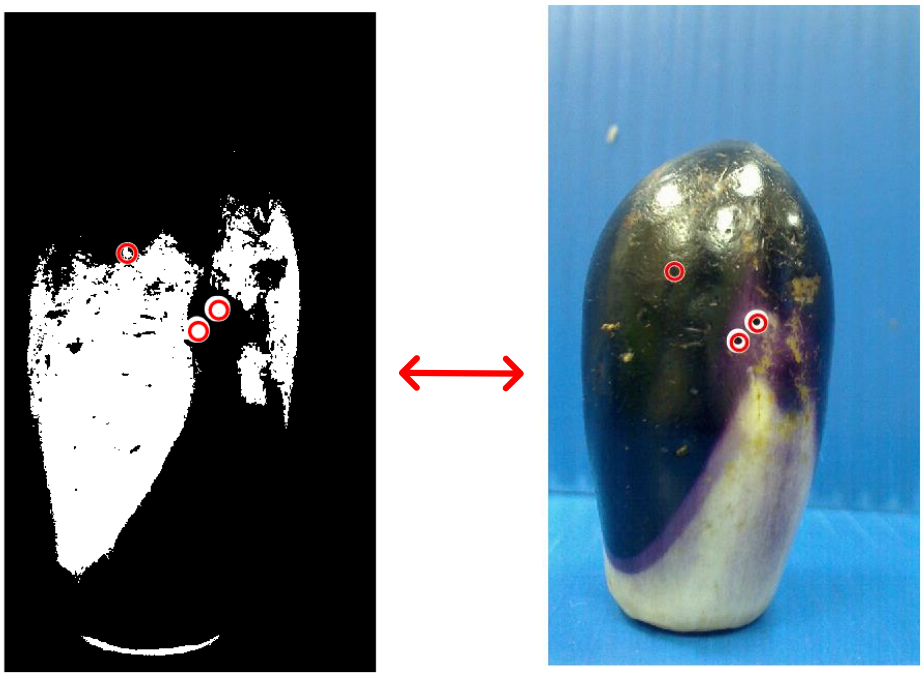
1.ค่าสัมประสิทธิ์ในการแปลงรูป Threshold โดยถ้าค่ามากหรือน้อยไปก็อาจทำให้ได้จุดที่ไม่ใช่รูหนอน หรือ ได้ไม่ครบทุกจุด เช่นดังรูป ปรับ เป็น 0.27 จะได้เพียงจุดเดียว



2.การกำหนดขนาดของวงกลมที่นำมาคำนวน โดยจะถ้าขนาดใหญ่หรือเล็กเกินอาจทำให้ได้ ตำแหน่งที่ผิดพลาดได้ เช่น ดังรูปกำหนด Circle_Size = 3 และ 5



5.ถ้ากำหนดจำนวนเปอร์เซ็นของการเท่ากันน้อยเกินไปอาจทำให้ได้ ตำแหน่งที่ผิดพลาดเช่นกัน ดังรูป กำหนด เป็นมากกว่า 0.8 ของทั้งหมด



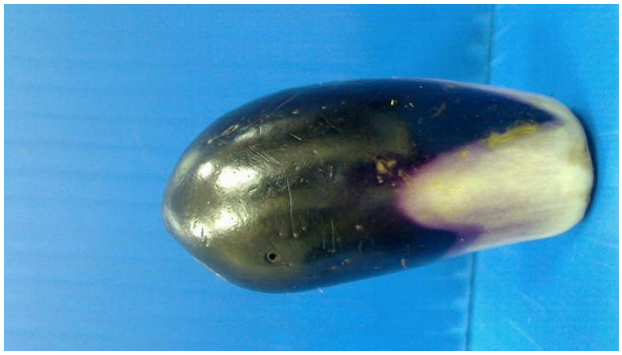
จะเห็นว่าเพียงลดมาจากค่าเดิม 0.7 ก็ทำให้ได้ข้อผิดพลาดมา 1 จุดแล้ว

4.การคำนวณหาตำแหน่งทับซ้อน กรณีที่ รูปหนอนทับกันหรือรูหนอนอยู่ติดกันมาก ๆ อาจทำให้โปรแกรมเลือกเพียง 1 รูเท่านั้นแทนที่จะเลือกทั้ง 2 เพราะรูหนอนอยู่ใกล้กันเกินไป

Conclusion

จากการค้นคว้าทำการทดลองจะเห็นว่าได้ผลลัพธ์ตามที่ต้องการ แต่ถ้าเปลี่ยนรูป Input ที่มีค่าแสงต่างจาก 2 รูปข้างต้น ก็จะต้องมากำหนดค่า สัมประสิทธิ์การแปลงรูปใหม่อีก และ ขึ้นอยู่กับรูปอีกเช่นกันว่าอยู่ใกล้หรืออยู่ไกลโดยถ้าอยู่ไกลก็มีโอกาสที่ รูหนอนจะมีขนาดเล็กลงกว่าปกติได้ จึงต้องตั้งค่า ขนาดของวงกลมที่จะนำมา Matching กับรูปด้วย และอาจจะมีลอยฟกช้ำบนตัวมะเขือม่วงหรือ Object ที่นำมาหารูหนอนก็อาจจะทำให้เกิดการคำนวณที่ผิดพลาดได้ด้วยเช่นกัน

ภาคผนวกรูปภาพ



(ก) WormHole_1H

(ข) WormHole_1H

ภาคผนวก

```
%ima = imread('WormHole_1H.tif');
ima = imread('WormHole_2H.tif');

imax = imrotate(ima,-90);
ima_gray = rgb2gray(ima);
ima_gray = imrotate(ima_gray,-90);
ima_gray = imbinarize(ima_gray,0.23);
ima_gray = ~ima_gray;

[X , Y] = size(ima_gray);
centers = [];
pad_size = 0;

for circle_size = [13 15 17 19 21 23]

    Z = zeros(circle_size);
    origin = [round((size(Z,2)-1)/2+1) round((size(Z,1)-
1)/2+1)];
    radius = round(sqrt(numel(Z)/(2*pi)));
    [Cx,Cy] = meshgrid((1:size(Z,2))-
origin(1),(1:size(Z,1))-origin(2));
    Z(sqrt(Cx.^2 + Cy.^2) <= radius) = 1;

    kernel = Z;
    [x,y] = size(Z);
    ima_gray_pad = padarray(ima_gray,[(x-1)/2 (y-
1)/2],0,'both');

    count = 0;
    redundance = 0;
    old_redundance = 0;
    for i = 1 : X
        for j = 1 : Y
            avg = 0;
            for k = 1  : x
                for l = 1 : y
                    if ima_gray_pad(i+k-1,j+l-1) ==
kernel(k,l)
                        avg = avg+1;
                    end
                end
            end
```

```
                if  avg > 0.87*((x*y))
                    if 0.95*(x*y) > avg
                         count = count +1;
                         if count > 0
                             pad_size = circle_size;
                         end
                             centers =
[centers;j+(circle_size-pad_size)/2 i+(circle_size-
pad_size)/2];
                    end
                end
            end
        end
        if count > 0
            pad_size = circle_size;
        end
end

    figure;
    imshow(imax);
    [n o] = size(centers);
    real_centers = [];
    index = 0;
    for kk = 1 : n
        if index == 0
            index = centers(1,1);
            real_centers = [real_centers;centers(1,1)
centers(1,2)];
        else
            if abs(centers(kk,1)-index) > circle_size/2
                index = centers(kk,1);
                real_centers = [real_centers;centers(kk,1)
centers(kk,2)];
            end
        end
    end

    [n o] = size(real_centers);
    for m = 1 : n
        viscircles(real_centers(m,1:2),pad_size/2);
    end
```