การมองเห็นของคอมพิวเตอร์ขั้นพื้นฐาน

Fundamentals of Computer Vision

ความรู้เบื้องต้นเกี่ยวกับภาพและการประมวลผลภาพ

ภิญโญ แท้ประสาทสิทธิ์

(pinyo at su.ac.th, pinyotae at gmail dot com)

ภาควิชาคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยศิลปากร

สัปดาห์ที่ 2

เรื่องที่ต้องการศึกษา

- ความรู้เบื้องต้นเกี่ยวกับภาพและการจัดการภาพ
- พื้นฐานการจัดการภาพในระดับพิกเซล
  - โมเดลสีพื้นฐานและการเปลี่ยนภาพสีให้เป็นภาพเฉดเทา
  - การทำเธรสโฮลด์แบบพื้นฐาน
  - การทำเธรสโฮลด์ด้วยวิธีของโอซึ (Otsu's Thresholding)
  - Adaptive and Advanced Thresholding
- พื้นฐานการเขียนโปรแกรมจัดการภาพ
  - การโหลดและแสดงรูป
  - การบันทึกภาพ
  - การแก้ไขค่าในรูปภาพ

ความรู้เบื้องต้นเกี่ยวกับภาพ: รูปแบบของภาพที่พบบ่อย

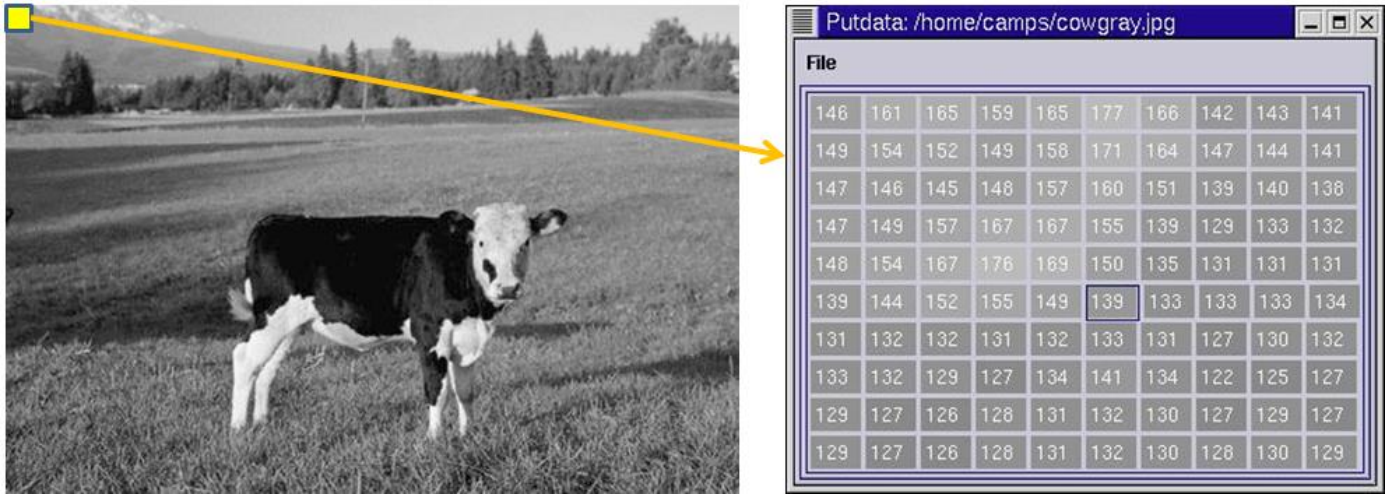
- ภาพที่พบบ่อยมักจะเป็นภาพสองมิติ
  - มักได้มาจากกล้องถ่ายรูปทั่วไป หรือกล้องโทรศัพท์มือถือ/แท็บเบล็ต
  - หรือตัดมาจากเฟรมในวีดีโอ (ภาพมักจะเบลอเมื่อหยุดเฟรม)
- โดยทั่วไปภาพที่อยู่ในไฟล์จะถูกบีบอัดไว้
  - วิธีในการบีบอัดภาพนั่นแหละที่ทำให้ภาพมีหลายรูปแบบ เช่น JPEG, PNG, TIFF และ Bitmap เป็นต้น
  - มีทั้งภาพสีและเฉดเทา (หลายคนเรียกว่าขาวดำ)
  - บางทีภาพเฉดเทาก็ถูกบันทึกไว้เป็นแบบภาพสี (ทำแบบนี้ได้ แต่มักจะเสียพื้นที่ในดิสก์เพิ่ม)
  - การบีบอัดอาจจะทำให้ภาพเสียความคมชัดได้ถ้าเลือกใช้วิธีแบบ Lossy เช่น JPEG หรืออาจจะคงความคมชัดไว้ได้เช่นเดิม เช่น Lossless

ความรู้เบื้องต้นเกี่ยวกับการจัดการภาพ

- ภาพที่พบบ่อยเป็นภาพในไฟล์ที่ถูกบีบอัดไว้ก็จริง
- แต่พอเราจะจัดการกับภาพเราจะโหลดภาพเข้ามาในหน่วยความจำ
- โดยทั่วไปภาพที่อยู่ในหน่วยความจำจะ**ไม่**ถูกบีบอัดไว้
  - แสดงว่าตอนอยู่ในไฟล์กับตอนอยู่ในหน่วยความจำมันมีรูปแบบที่ไม่เหมือนกัน
  - ตอนอยู่ในไฟล์เราเน้นเรื่องประหยัดพื้นที่ ได้ดุลระหว่างความชัดกับขนาดไฟล์
  - ตอนอยู่ในหน่วยความจำ เราต้องทำให้เขียนโปรแกรมได้ง่าย คำนวณได้เร็ว
- เพื่อให้เขียนโปรแกรมได้ง่าย เราจึงจัดภาพให้อยู่ในรูปแบบที่เข้าใจง่ายด้วย
  - จัดแบบอาเรย์สองมิติจะเข้าใจง่าย
  - ตำแหน่งของช่องข้อมูลในอาเรย์ ตรงกับจุด (พิกเซล) ในภาพ
  - แต่ว่าค่าแต่ละค่าในอาเรย์นั้นมันแสดงถึงอะไร ทำไมมันเป็นสีหรือเฉดเทาได้

ภาพเฉดเทาและภาพขาวดำ

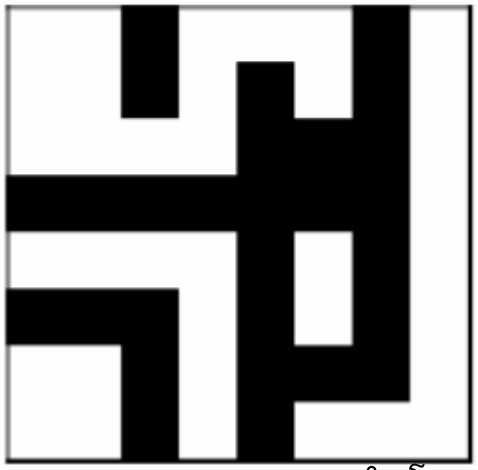
- ภาพเฉดเทาและภาพขาวดำต่างกันตรงที่ปริมาณระดับความสว่างที่แสดงได้
  - ภาพเฉดเทามักจะแสดงระดับความสว่างได้ 256 แบบ
  - ค่าพิกเซลในภาพเฉดเทามักมีค่าตั้งแต่ 0 ถึง 255 (0 = ดำ, 255 = ขาว)
  - บางทีก็มีค่าติดลบหรือค่าที่เกินกว่า 255 แต่มักจะมีปัญหากับการแสดงผล
  - ภาพขาวดำแสดงระดับความสว่างได้สองแบบ (binary) คือขาวกับดำสนิท
  - ค่าพิกเซลในภาพขาวดำมักจะเป็น 0 กับ 1 หรือไม่ก็ 0 กับ 255 (ทราบหรือไม่ว่ามีเหตุผลอย่างไรกับการเก็บข้อมูลสองแบบนี้)



Slide credit: Robert T. Collins and Kristen Grauman

ลึก ๆ กับการจัดการภาพขาวดำ

- เราต้องการแสดงผลภาพที่มีเฉพาะสีขาวกับสีดำเพียงสองสีบนจอภาพ
  - แต่วิธีการแสดงผลมักจะสนับสนุนแต่ภาพเฉดเทา
  - เพื่อให้ใช้วิธีแสดงผลเดิมได้ เรามักแทนค่าพิกเซลในภาพขาวดำด้วย 0 กับ 255
  - การแสดงผลด้วย 0 กับ 255 ใช้พื้นที่เก็บข้อมูลมากกว่าแบบ boolean หรือไม่?
  - ภาษาโปรแกรมทั่วไป เช่น C++ เก็บข้อมูลบูลีนด้วยพื้นที่ 1 ไบต์
    → ใช้พื้นที่เท่ากับ unsigned char (ในจาวาไม่มี และเรามักจะใช้ int แทน)
  - แต่ในการอธิบายยังไงเขียนเลข 0 กับ 1 ก็ง่ายกว่าเขียนด้วย 0 กับ 255



1	1	0	1	1	1	0	1
1	1	0	1	0	1	0	1
1	1	1	1	0	0	0	1
0	0	0	0	0	0	0	1
1	1	1	1	0	1	0	1
0	0	0	1	0	1	0	1
1	1	0	1	0	0	0	1
1	1	0	1	0	1	1	1

ภาพขาวดำ โดยทั่วไป 1 คือสีขาวและ 0 คือสีดำ

Slide credit: Kristen Grauman

เราเก็บอะไรไว้ในภาพขาวดำ

- Foreground and background
  - ➢ บางทีสีดำก็เป็น foreground ส่วนสีขาวเป็น background
- Regions of interest (พื้นที่ที่เราสนใจ / พื้นที่เป้าหมาย)

ป้ายทะเบียนรถทั่วไปสามารถเก็บเป็นภาพขาวดำได้ดี

การประยุกต์ใช้ภาพขาวดำเพื่อการตรวจสอบทางอุตสาหกรรม

Industrial Inspection

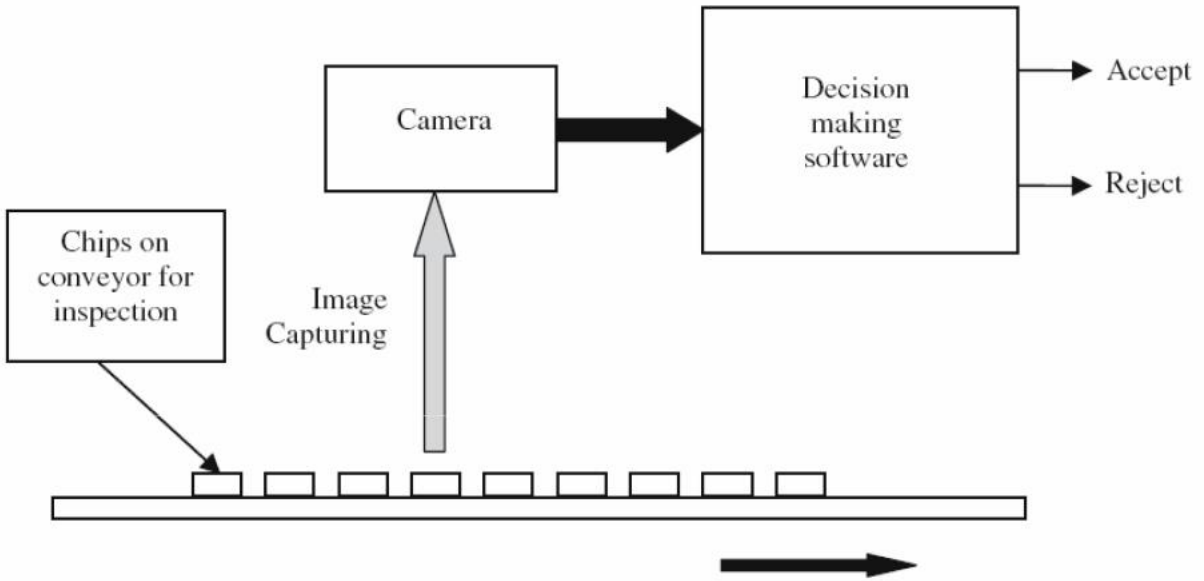


Fig. 3 Schematic diagram of marking inspection setup at Texas Instruments

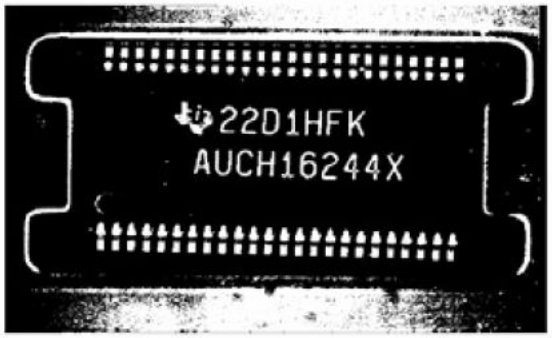
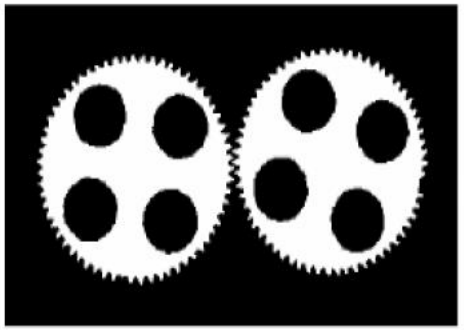


Fig. 7 Binarized image

Fig. 9 Row sum for separating a row

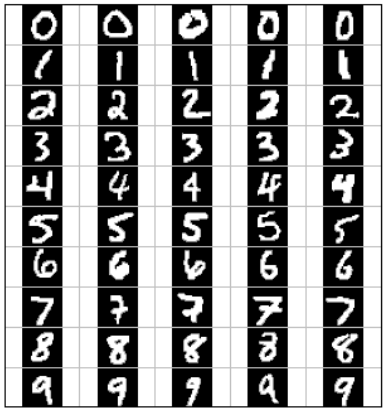


R. Nagarajan et al. “A real time marking inspection scheme for semiconductor industries“, 2006

Slide credit: Bastian Leibe

การใช้ภาพขาวดำเพื่อวิเคราะห์เอกสารหรือข้อความ

Document analysis / Text Recognition



การรู้จำลายมือ

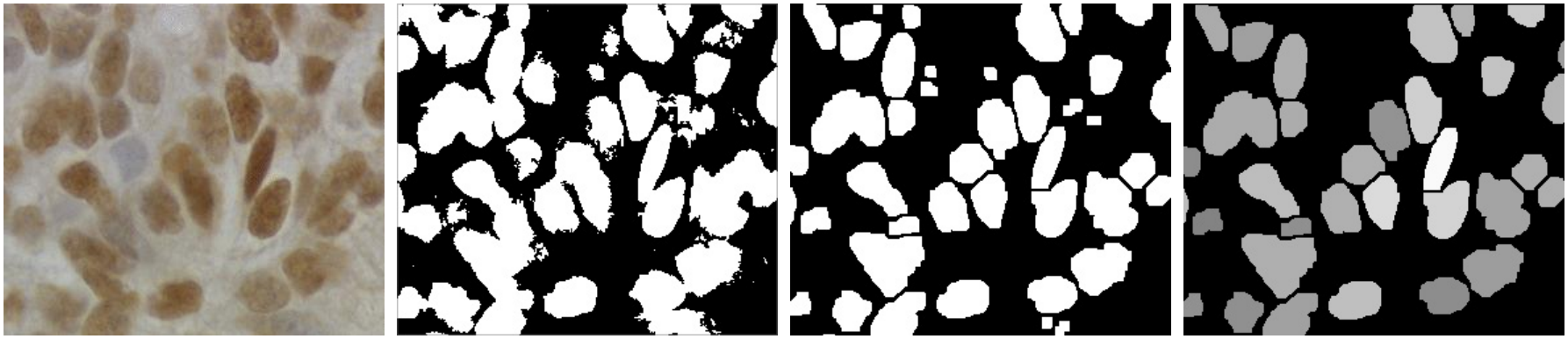
การวิเคราะห์เอกสารสแกน

การรู้จำข้อความในสภาพแวดล้อม (หลังจากการแยกข้อความมาได้)

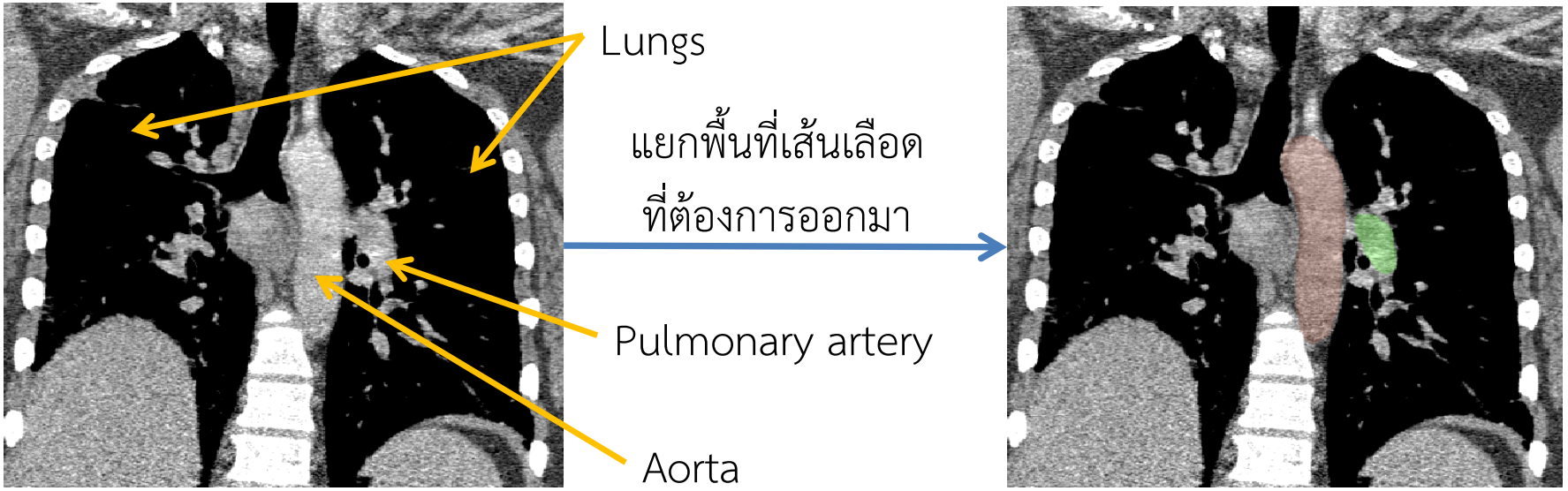
Source: Till Quack, Martin Renold

Slide credit: Bastian Leibe

การใช้ภาพขาวดำกับข้อมูลชีวภาพหรือด้านการแพทย์



Source: D. Kim *et al*., Cytometry 35(1), 1999, Bastian Leibe



รหัสผู้ป่วย 20349.3.7

Tree Analyzer 3.0 by Yu, Taeprasartsit, Wan, and Higgins 2011

แล้วภาพสีล่ะ จัดเก็บในอาเรย์อย่างไร

- เวลาที่เรากล่าวถึงภาพสี เรามักจะกล่าวถึงการเก็บภาพตามแม่สีแสง
  - คือสีแดง เขียว และ น้ำเงิน เรามักเรียกติดปากว่าสี RGB (มาจาก Red Green Blue)
  - การเลือกเก็บสีตามแม่สีแสง เป็นวิธีที่นิยมที่สุด (แต่ใช่วิธีเดียว)
  - จอภาพรองรับการแสดงผลจากแม่สีแสงโดยตรง ทำให้ง่ายต่อความเข้าใจไปอีกระดับ
- แต่ในเมื่อมีสามสีที่ต้องเก็บ แล้วจะเก็บอย่างไรดี
  - ทางเลือกในการเก็บมีหลายทาง ทางที่นิยมที่สุดก็คือแบ่งภาพออกเป็นช่องสัญญาณ
  - ในที่นี้ก็จะ “ถือว่า” มีช่องสัญญาณ RGB สามช่อง
  - แต่ละช่องมีตัวเลขอยู่ในช่วง 0 ถึง 255 คล้ายภาพเฉดเทา (เก็บด้วยตัวแปร 8 บิตได้)
  - ดังนั้นเวลาจัดเก็บพิกเซลอันหนึ่ง ก็จะใช้พื้นที่ 3 ไบต์
  - ระบบซอฟต์แวร์จำนวนมากจึงจับข้อมูลจากช่องสัญญาณรวมกันไว้ใน int ตัวเดียว
  - การเก็บจึงมักเป็นอาเรย์สองมิติเช่นเดิม แต่ใช้การตัดแบ่ง int เป็นช่องสัญญาณแทน

Slide credit: Kristen Grauman

แล้วโมเดลสีอื่น ๆ มีไว้ทำไม

- โมเดลสีอื่น ๆ มีอยู่มากมาย แต่ละโมเดลก็มีประโยชน์ใช้สอยต่างกันไป
- เช่นโมเดลสีแบบ HSV จะแยกเรื่องความเป็นสี กับความสว่างออกจากกัน
  - มีความใกล้เคียงกับการมองเห็นของมนุษย์มากขึ้น
  - ใช้ในการจำแนกสีที่อยู่ในภาพได้ดีกว่า เพราะตัดทอนความสว่างออกไปเป็นอีกช่องสัญญาณ (ช่อง V) [เราจะพูดเรื่องนี้อย่างละเอียดในภายหลัง]
- โมเดลสีที่มีงานเจาะจงอย่าง TSL ช่วยให้ตรวจหาพื้นที่ผิวหนังมนุษย์ในภาพได้อย่างรวดเร็วและแม่นยำ
- โมเดลสีแบบ CMYK ใช้กับการบริหารสีในเครื่องพิมพ์
- โมเดลสีแบบ CIE L*a*b* และ CIEDE2000 จะสอดคล้องกับประสาทสัมผัสของมนุษย์มากเป็นพิเศษ

รู้สึกโมเดลสีมีเยอะ แบบนี้เลือกใช้อะไรดี

- จะเลือกใช้โมเดลอะไรต้องดูที่การใช้งานประกอบกันไปเสมอ
- เช่น โมเดลสีที่สอดคล้องกับการมองเห็นของเราอย่าง CIEDE2000 ช่วยได้หลายงานแต่ก็ใช้พลังในการประมวลผลสูง เป็นต้น
- อย่างไรก็ตาม ในระดับพื้นฐานการวิเคราะห์ภาพเฉดเทาหรือขาวดำก็ช่วยเราได้มากพอตัว แม้แต่ภาพทางการแพทย์จำนวนมากก็เป็นภาพเฉดเทา

✓ ดังนั้นตอนนี้อย่าเพิ่งคิดอะไรมาก เรามาจัดการภาพเฉดเทาให้ได้คล่อง ๆ ก่อนแล้วค่อยมาคิดเรื่องภาพสีในภายหลัง

เรื่องที่ต้องการศึกษา

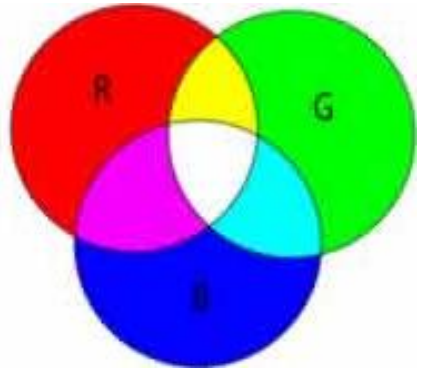
- ความรู้เบื้องต้นเกี่ยวกับภาพและการจัดการภาพ
- **พื้นฐานการจัดการภาพในระดับพิกเซล**
  - **โมเดลสีพื้นฐานและการเปลี่ยนภาพสีให้เป็นภาพเฉดเทา**
  - การทำเธรสโฮลด์แบบพื้นฐาน
  - การทำเธรสโฮลด์ด้วยวิธีของโอซึ (Otsu's Thresholding)
  - Adaptive and Advanced Thresholding
- พื้นฐานการเขียนโปรแกรมจัดการภาพ
  - การโหลดและแสดงรูป
  - การบันทึกภาพ
  - การแก้ไขค่าในรูปภาพ

โมเดลสีพื้นฐาน

- เนื่องจากเราจะจัดการภาพเฉดเทาก่อน แต่ภาพที่ได้มาดันเป็นภาพสี
- ดังนั้นมาทำความเข้าใจกันก่อนเลยว่าเราจะแปลงภาพสีกันอย่างไร
- ในตอนนี้เอาแค่แนวคิด และสมมติว่าภาพที่เก็บไว้เป็นแบบ RGB (นิยมสุด)
- ในโมเดลสี RGB เราใช้ค่าตัวเลข แสดงความสว่างของสีนั้น ๆ
  - เช่น ถ้าค่า R มีค่า 255 เราก็จะได้สีแดงสว่างไสว
  - แต่สีที่เห็นเกิดจากการผสมค่า R G B เข้าด้วยกัน ดังนั้นต้องคำนึงถึงสีอื่น ๆ ด้วย
  - ถ้าค่า G และ B มีค่าเป็นศูนย์ ก็แสดงว่าสีสองสีนั้นไม่มีอิทธิพลใด ๆ เลย ดังนั้น ถ้า R = 255, G = 0 และ B = 0 เราก็จะได้สีแดงสดสว่างไสว
  - ถ้าค่า R G และ B มีค่าเท่ากันหมด เช่น R = G = B = 255 เราก็จะได้สีขาว

ทำไมเป็นสีขาว

- การผสมสี RGB นั้นเป็นไปตามกลไกของแม่สีแสง ดังนั้นผลของการผสมสีจะมีลักษณะเดียวกับภาพทางด้านใต้นี้



- ถ้าเราปรับค่า R, G, B ไปเป็นรูปแบบต่าง ๆ ในช่วง 0 ถึง 255 ผลที่ได้ก็จะต่างกันไป ได้สีออกมาหลากหลายรูปแบบ
- เช่นสีเทาก็คือ R = G = B = 127 เป็นต้น

มาแปลงภาพสีเป็นเฉดเทากันเถอะ

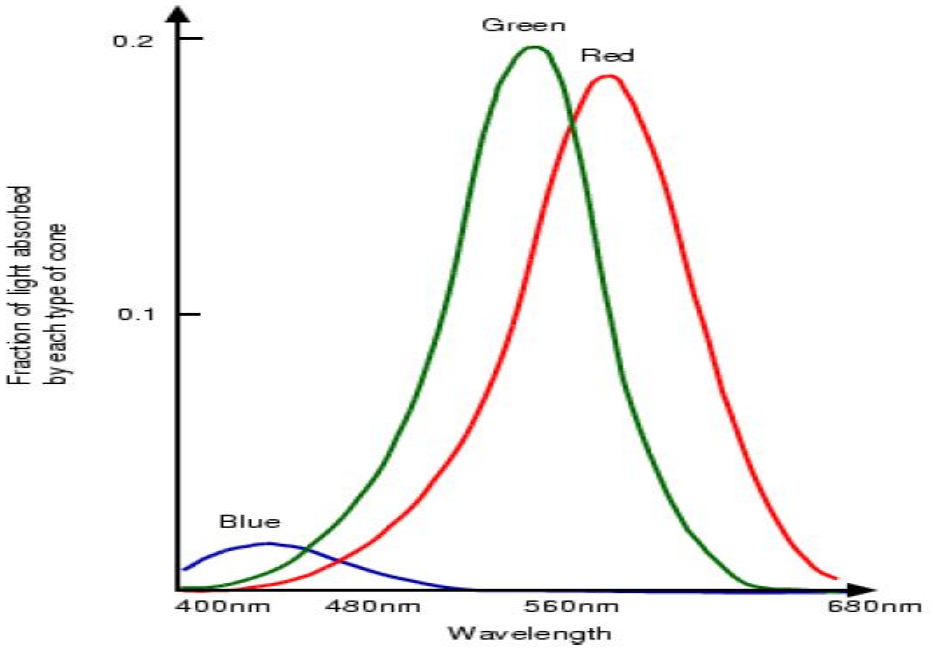
- เรารู้แล้วว่าเรามีค่า R, G และ B อยู่ในช่วง 0 ถึง 255 ค่ายิ่งมากก็ยิ่งสว่าง
- ในขณะที่ภาพเฉดเทาก็อยู่ในช่วงเดียวกัน และบ่งบอกถึงความสว่าง
- ดังนั้นเราต้องมีวิธีในการนำค่า R, G และ B มารวมกันแล้วแปลงเป็นความสว่างให้ได้
  - วิธีตื้น ๆ และผิดก็คือจับค่าพวกนี้มาหารเฉลี่ยกันตรง ๆ คือ

    Gray = $\frac{R+B+G}{3}$
  - วิธีที่ดูดีขึ้นมากแต่ก็ยังผิดจากที่ควรเป็นก็คือคิดแบบ Euclidean

    คือใช้การคำนวณว่า Gray = $\sqrt{\frac{R^2+G^2+B^2}{3}}$
  - สาเหตุหลักของความผิดพลาดเหล่านี้ก็คือการให้น้ำหนักทั้งสามแม่สีเท่ากัน

อ๋า ให้ค่าน้ำหนักเท่ากันก็ยุติธรรมดีนี่ ทำไมผิดอ่ะ

- ที่ผิดก็เพราะมันไม่สอดคล้องกับประสาทสัมผัสของมนุษย์ทั่วไปนั่นเอง
- จำได้หรือไม่ว่าทำไม Night Vision Goggles จึงเลือกสีเขียว



- จากภาพ แสดงว่าสายตาของคนทั่วไปให้น้ำหนักสีเขียวกับแดงสูงมาก

ถ้างั้นสูตรคำนวณที่ควรจะเป็นนั้นมีหน้าตาอย่างไร

- ก่อนอื่นต้องเข้าใจก่อนว่า สายตาแต่ละคนไม่เหมือนกัน ความไวในการรับสีแต่ละคนก็ไม่เท่ากัน แต่แนวโน้มต่าง ๆ ของคนทั่วไปอยู่ในทางเดียวกัน
- ดังนั้นค่าน้ำหนักที่ใส่เข้าไป ใช่ว่าจะถูกต้องตลอดกาล อาจจะมีการวิจัยมาหักล้างตัวเลขพวกนี้ได้
- ยิ่งไปกว่านั้น การรับรู้ของสายตาเราเป็นแบบไม่เชิงเส้น (Non-linear) การใช้สูตรแบบเชิงเส้น (Linear) ก็มีความผิดพลาด
- แต่ถึงกระนั้น สายตาเราก็มักจะไม่คมพอที่จะแยกความแตกต่างได้มากนักในหลาย ๆ กรณีเราจึงยอมรับว่าสามารถใช้สูตรแบบเชิงเส้นว่า

$$\text{Gray} = 0.2126\ R + 0.7152\ G + 0.0722\ B$$

แล้วตอนเขียนโปรแกรมล่ะ

- มันตรงตรงมามากเลย สมมติว่าในจุดสี (พิกเซล) ที่เรากำลังพิจารณาอยู่
  - เราสามารถอ่านค่าสีของมันออกมาเก็บไว้ในตัวแปร int ได้เป็น r, g, b
  - เราต้องการเก็บผลการแปลงไว้ในตัวแปร int ชื่อ gray เราก็เขียนไปเลยว่า

```
gray = 0.2126 * r + 0.7152 * g + 0.0722 * b;
```

- ทว่าภาพมันไม่ได้แค่พิกเซลเดียว แต่มันมีได้เป็น 10 ล้านหรือมากกว่า
- ในทางปฏิบัติเราจึงวนลูปคำนวณพวกนี้ไปทีละพิกเซลจนหมด
  - เนื่องจากภาพเป็นอาเรย์สองมิติ เรามักจะวนลูปสองชั้นไปด้วย
- ว่าแต่แค่เริ่มเรียน เราก็ต้องออกแรงแปลงภาพกันอย่างนี้แล้วเหรอ
  - ที่จริงไม่ต้องขนาดนั้น เพราะเรามีคำสั่งแปลงพวกนี้ติดมาตั้งแต่แรก สั่งโหลดภาพสีแล้วให้มันจัดหน่วยความจำแบบเฉดเทาได้เลย สบายมาก

แล้วสรุปเราเรียนเรื่องแปลงภาพสีเป็นเฉดเทาทำไมเนี่ย

- เพราะมันเป็นการสร้างความคุ้นเคยกับระบบหน่วยความจำของภาพสีและเฉดเทาที่เข้าใจได้ง่ายที่สุด
  - ถ้าเริ่มปุ๊บก็เขียนกับเรื่องยาก ๆ เลยโดยปรกติจะทำไม่ค่อยได้
  - มันเป็นแบบฝึกหัดสำหรับสร้างพื้นฐานที่ดีมาก
- มีโมเดลสีที่มีประโยชน์มากอย่าง TSL และ CIEDE2000 แต่ยังไม่มีฟังก์ชันสำหรับแปลงจาก RGB ไปเป็นสองแบบนี้ ถ้าเราจะใช้เราก็ต้องหัดจากพื้นฐานตรงนี้แล้วต่อยอดขึ้นไป

เรื่องที่ต้องการศึกษา

- ความรู้เบื้องต้นเกี่ยวกับภาพและการจัดการภาพ
- พื้นฐานการจัดการภาพในระดับพิกเซล
  - โมเดลสีพื้นฐานและการเปลี่ยนภาพสีให้เป็นภาพเฉดเทา
  - **การทำเธรสโฮลด์แบบพื้นฐาน**
  - การทำเธรสโฮลด์ด้วยวิธีของโอซึ (Otsu's Thresholding)
  - Adaptive and Advanced Thresholding
- พื้นฐานการเขียนโปรแกรมจัดการภาพ
  - การโหลดและแสดงรูป
  - การบันทึกภาพ
  - การแก้ไขค่าในรูปภาพ

การทำเธรสโฮลด์แบบพื้นฐาน

- การทำเธรสโฮลด์ (Thresholding) คือการพิจารณาค่าและเปรียบเทียบกับค่าจุดแบ่ง (Threshold หรือ Threshold Value) แล้วตัดสินประเภท
  - เรื่องหลัก ๆ ของการแบ่งประเภทก็คือการจัดทำภาพเป็นแบบขาวดำ
  - โดยส่วนใหญ่ เราจะเลือกค่ามาค่าหนึ่ง ถ้าค่าเฉดเทามีค่าสูงกว่าค่าที่เลือกมา เราก็จะแบ่งประเภทว่าจุดค่าเฉดเทาดังกล่าวเป็นสีขาว ถ้าไม่มากกว่าก็เป็นสีดำ
  - การแบ่งประเภทยังมีอีกหลายแบบ เช่นในภาพ PET (Positron Emitted Tomography) บางประเภท จุดที่มีปฏิกิริยาเคมีบางอย่างสูงจะมีความสว่างมากในภาพ PET
  - ถ้าจุดดังกล่าวอยู่ในพื้นที่ปอด บริเวณนั้นก็มักจะเป็นมะเร็งปอด เป็นต้น
  - ดังนั้นการแบ่งประเภทจึงเป็นเรื่องของการใช้งาน รวม ๆ แล้วเราเรียกการแบ่งประเภทจากค่าที่เลือกมาว่า Thresholding

การสร้างภาพขาวดำแบบง่าย (Simple Image Binarization)

ใช้ Basic Thresholding

- Gray or color image ➔ binary mask
- มีใช้กันอยู่หลายแบบ แต่ที่เจอบ่อย ๆ คือ
  - แบบเปรียบเทียบค่าด้านเดียว (one-sided comparison)

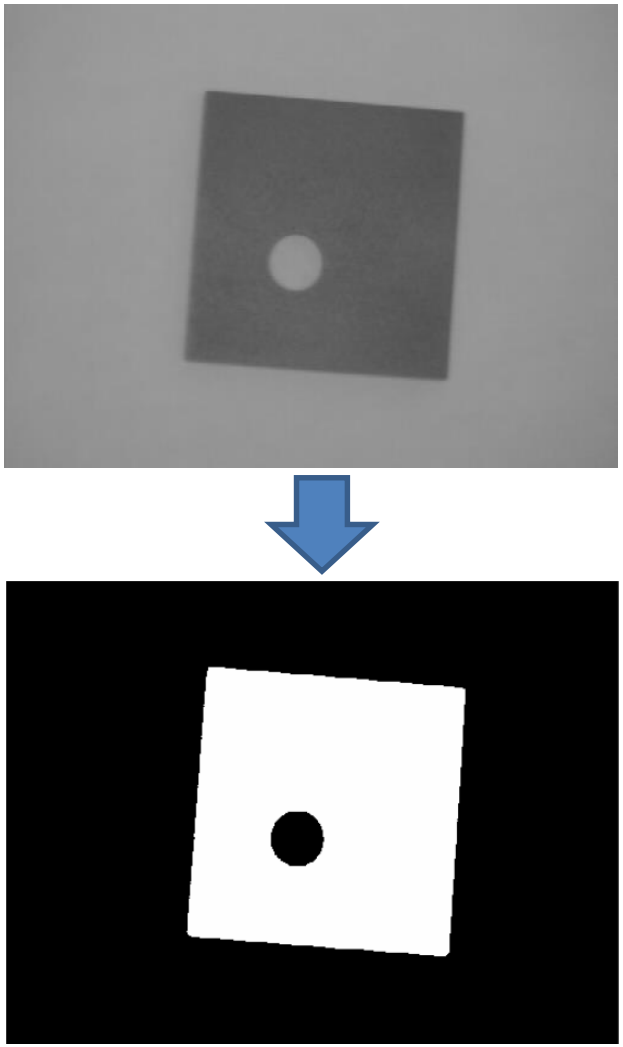
$$I_T(x,y) = \begin{cases} 1, & \text{if } I(x,y) \geq T \\ 0, & \text{otherwise} \end{cases}$$

  - แบบเปรียบเทียบค่าสองด้าน (two-sided comparison)

$$I_T(x,y) = \begin{cases} 1, & \text{if } T_1 \geq I(x,y) \geq T_2 \\ 0, & \text{otherwise} \end{cases}$$

  - แบบสมาชิกเซ็ต (set membership)

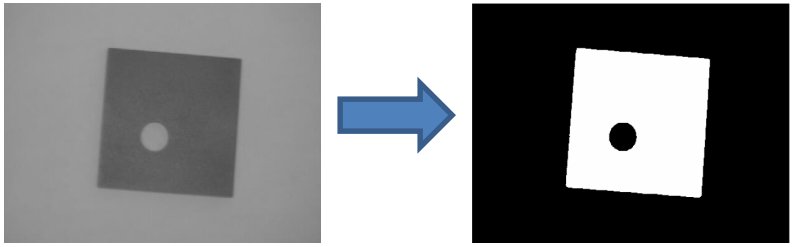
$$I_T(x,y) = \begin{cases} 1, & \text{if } I(x,y) \in A \\ 0, & \text{otherwise} \end{cases}$$



Slide credit: Bastian Leibe, Image source: Image Processing Learning Resources, http://homepages.inf.ed.ac.uk/rbf/HIPR2/images/wdg2.gif

การเลือกค่าเธรสโฮลด์

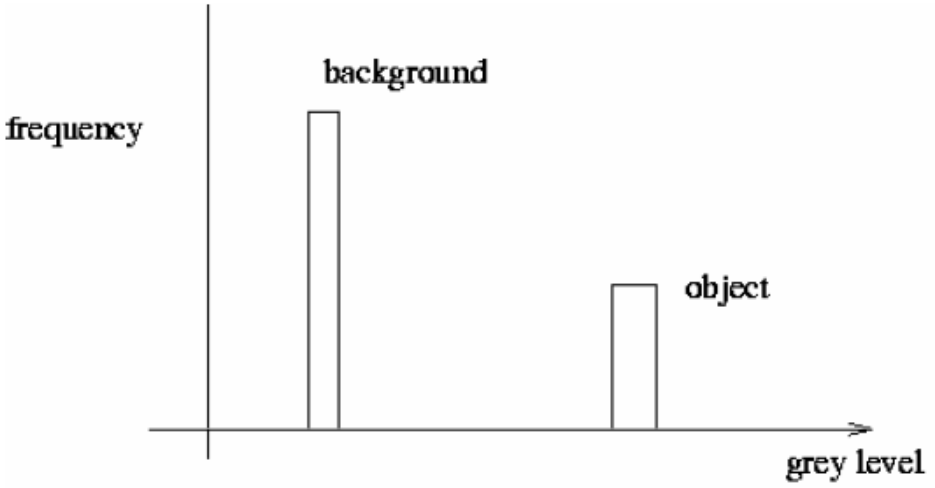
- การทำเธรสโฮลด์เป็นเรื่องที่ง่าย แต่การเลือกค่าเธรสโฮลด์ที่เหมาะสมเป็นงานที่ยาก
  - โดยปรกติแล้วเราต้องการแยกเอาวัตถุที่สนใจออกจากสิ่งอื่น ๆ



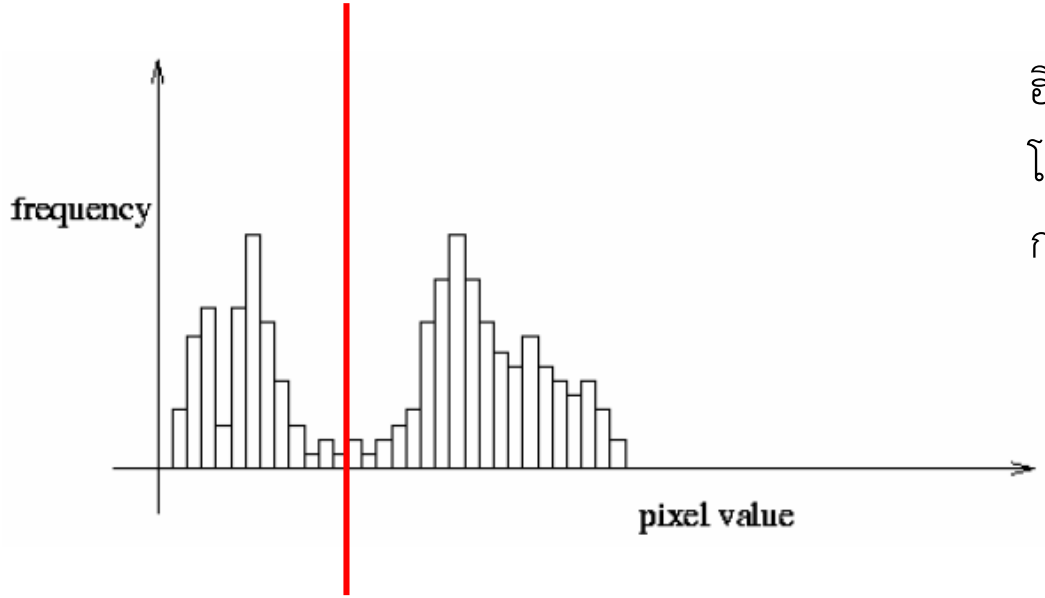
- พยายามแยกสิ่งที่ต้องการจากการกระจายตัวของค่าเฉดเทาที่ต่างกัน
  - ถ้าวัตถุเป้าหมายกับสิ่งที่อยู่รอบ ๆ มีความแตกต่างของค่าเฉดเทามาก → ง่าย
- วิธีที่พบบ่อย
  - แยกโดยการใช้ฮิสโทแกรมทวิฐานนิยม (bimodal histogram)
  - หาการกระจายตัวอิงพารามิเตอร์ที่ดีที่สุด (Fit a parametric distribution) เช่น การใช้ Mixture of Gaussian
  - ใช้ค่าเธรสโฮลด์พลวัตหรือค่าเธรสโฮลด์เฉพาะพื้นที่ (dynamic or local thresholds)

Slide credit: Bastian Leibe

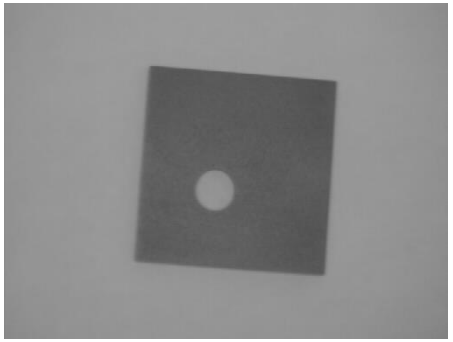
กรณีที่จัดการง่าย: Bimodal Intensity Histograms



ฮิสโทแกรมอุดมคติ,
วัตถุสีอ่อนวางอยู่บนพื้นหลังสีเข้ม

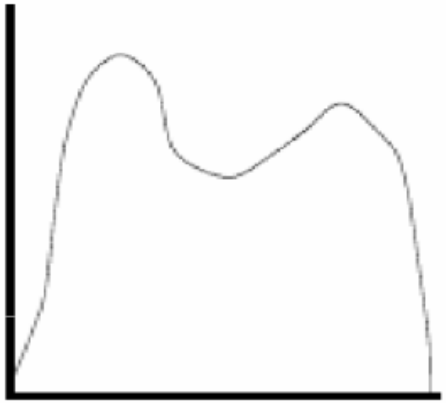


ฮิสโทแกรมที่มักจะเกิดขึ้นจริง,
โดยมากมีสัญญาณรบกวน หรือ
การกระจายของแสงในฉากไม่สม่ำเสมอ



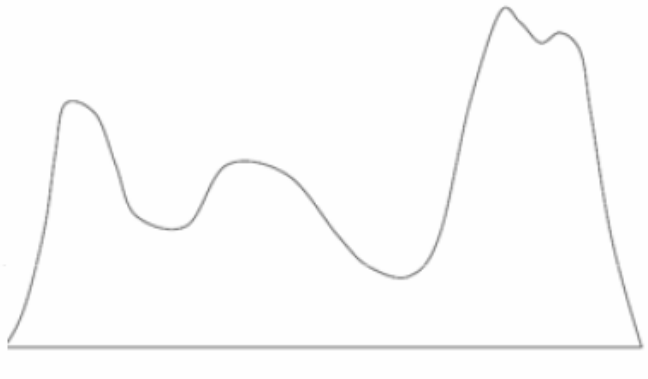
Source: Robyn Owens

แต่ว่าในชีวิตจริงนั้นชีวิตมันไม่ง่าย



Overlapping modes

ค่าเฉดเทาของวัตถุเป้าหมาย กับสิ่งอื่น ๆ มันทับซ้อนกัน (grayvalue overlap)



Multiple modes

ถ้าวัตถุเป้าหมายหรือพื้นหลังมีค่าเฉดเทาอยู่หลายกลุ่ม ก็เป็นการยากที่จะหาเธรสโฮลด์แบบอัตโนมัติ

การหาค่าเธรสโฮลด์เป็นเรื่องยากในกรณีทั่วไป → ต้องใช้ข้อมูลอื่นเป็นตัวช่วย

- ใช้ domain knowledge

  เช่น ถ้ารู้ว่าสิ่งที่ต้องการหาคือข้อความสีเข้มในเอกสาร และพื้นที่ในหน้ากระดาษส่วนใหญ่มีสีอ่อน

  → histogram quantile

  → วัตถุเป้าหมายแบ่งเป็นพื้นที่เล็ก ๆ จำนวนมาก

Source: Shapiro & Stockman

Slide credit: Bastian Leibe

แต่ว่าในชีวิตจริงนั้นชีวิตมันไม่ง่าย

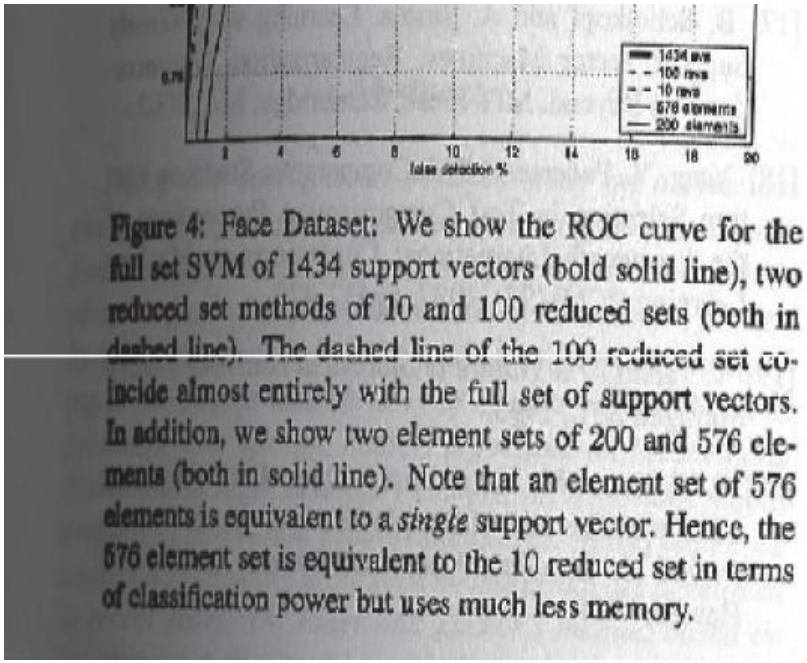
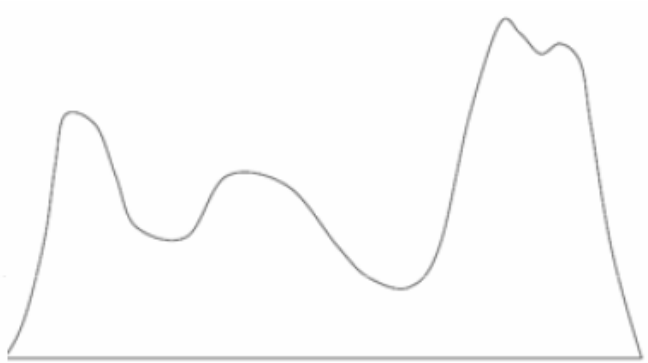


Figure 4: Face Dataset: We show the ROC curve for the full set SVM of 1434 support vectors (bold solid line), two reduced set methods of 10 and 100 reduced sets (both in dashed line). The dashed line of the 100 reduced set coincide almost entirely with the full set of support vectors. In addition, we show two element sets of 200 and 576 elements (both in solid line). Note that an element set of 576 elements is equivalent to a *single* support vector. Hence, the 576 element set is equivalent to the 10 reduced set in terms of classification power but uses much less memory.



Multiple modes

ถ้าวัตถุเป้าหมายหรือพื้นหลังมีค่าเฉดเทาอยู่หลายกลุ่ม ก็เป็นการยากที่จะหาเธรสโฮลด์แบบอัตโนมัติ

การหาค่าเธรสโฮลด์เป็นเรื่องยากในกรณีทั่วไป → ต้องใช้ข้อมูลอื่นเป็นตัวช่วย

- ใช้ domain knowledge

  เช่น ถ้ารู้ว่าสิ่งที่ต้องการหาคือข้อความสีเข้มในเอกสาร และพื้นที่ในหน้ากระดาษส่วนใหญ่มีสีอ่อนกว่า

  → histogram quantile

  → วัตถุเป้าหมายแบ่งเป็นพื้นที่เล็ก ๆ จำนวนมาก

Source: Shapiro & Stockman

Slide credit: Bastian Leibe

เรื่องที่ต้องการศึกษา

- ความรู้เบื้องต้นเกี่ยวกับภาพและการจัดการภาพ
- พื้นฐานการจัดการภาพในระดับพิกเซล
  - โมเดลสีพื้นฐานและการเปลี่ยนภาพสีให้เป็นภาพเฉดเทา
  - การทำเธรสโฮลด์แบบพื้นฐาน
  - **การทำเธรสโฮลด์ด้วยวิธีของโอซึ (Otsu's Thresholding)**
  - Adaptive and Advanced Thresholding
- พื้นฐานการเขียนโปรแกรมจัดการภาพ
  - การโหลดและแสดงรูป
  - การบันทึกภาพ
  - การแก้ไขค่าในรูปภาพ

Global Binarization [Otsu 1979]

- หาค่าเธรสโฮลด์ *T* ที่ทำให้ within-class variance (σ_{within}) ของคลาสทั้งสองที่ถูกแยกด้วย *T* มีค่าน้อยที่สุด (minimize within-class variance)

$$\sigma^2_{within}(T) = n_1(T)\sigma^2_1(T) + n_2(T)\sigma^2_2(T)$$

โดยที่

$$n_1(T) = |\{I(x,y) < T\}|, n_2(T) = |\{I(x,y) \geq T\}|$$

- วิธีนี้ให้ผลเหมือนกับการ maximize the between-class variance $\sigma_{between}$

$$\sigma^2_{between}(T) = \sigma^2 - \sigma^2_{within}(T)$$
$$= n_1(T)n_2(T)[\mu_1(T) - \mu_2(T)]^2$$

Nobuyuki Otsu (1979). "A threshold selection method from gray-level histograms". *IEEE Trans. Sys., Man., Cyber.* **9** (1): 62–66. doi:10.1109/TSMC.1979.4310076

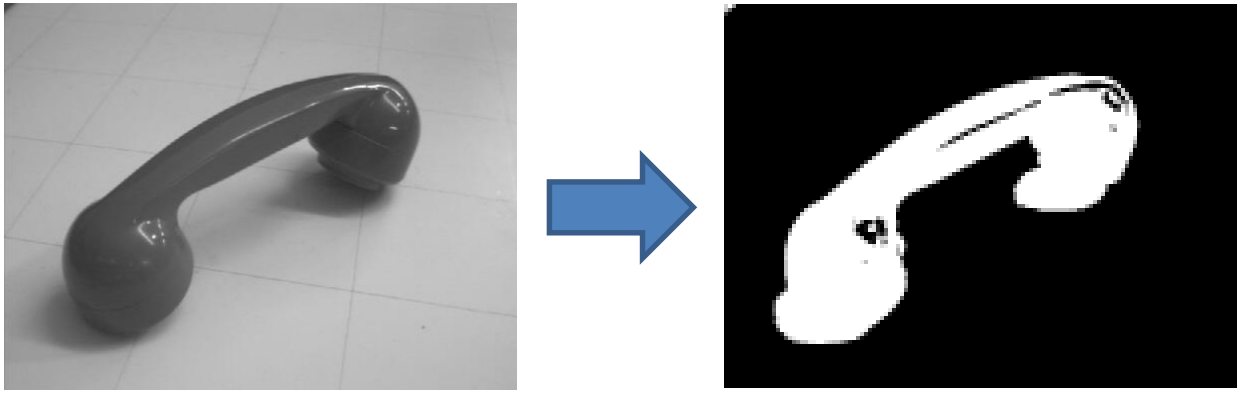
Slide credit: Bastian Leibe

Algorithm for Global Binarization

1. Precompute a cumulative grayvalue histogram *h*.
   (คำนวณฮิสโทแกรมค่าเฉดเทาสะสมเก็บไว้ก่อน—จะได้ใช้ซ้ำได้ไม่ต้องคำนวณอีก)

2. สำหรับแต่ละค่าเธรสโฮลด์ *T* ที่เป็นไปได้
   a) แยกพิกเซลออกเป็นสองกลุ่มตามค่า *T*
   b) หาค่า n_1 , n_2 ภายในฮิสโทแกรม *h* และคำนวณค่าเฉลี่ยเฉดเทาของแต่ละกลุ่ม
   c) คำนวณ $\sigma^2_{between}$

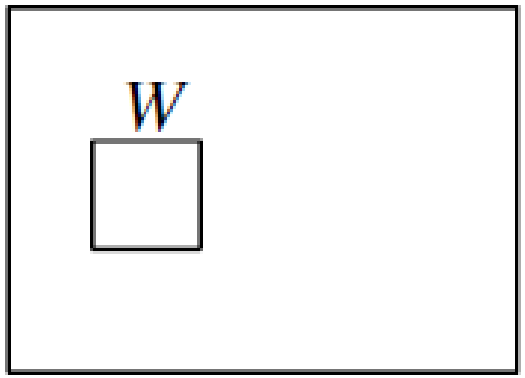
3. เลือกค่า $T^* = \arg\max_T [\sigma^2_{between}(T)]$



Slide credit: Bastian Leibe

Local Binarization [Niblack 1986]

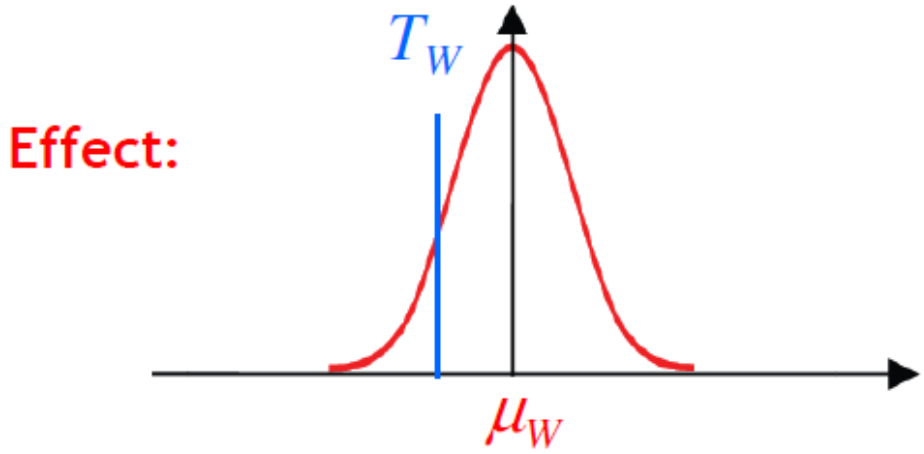
- หาค่าเธรสโฮลด์จากพื้นที่เล็ก ๆ ภาพในหน้าต่าง W
  (โดยทั่วไปคำว่าหน้าต่างในคอมพิวเตอร์วิชันจะหมายถึง
  พื้นที่เล็ก ๆ ที่กำลังพิจารณาอยู่)



$$T_W = \mu_W + k \cdot \sigma_W$$

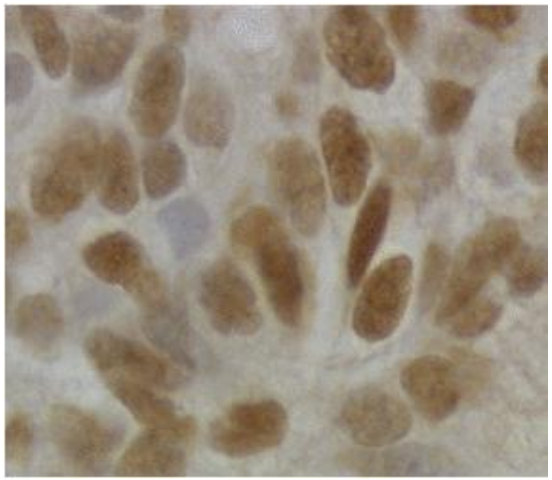
Standard deviation

โดยที่ $k \in [-1,0]$ เป็นพารามิเตอร์ที่ผู้ใช้กำหนด

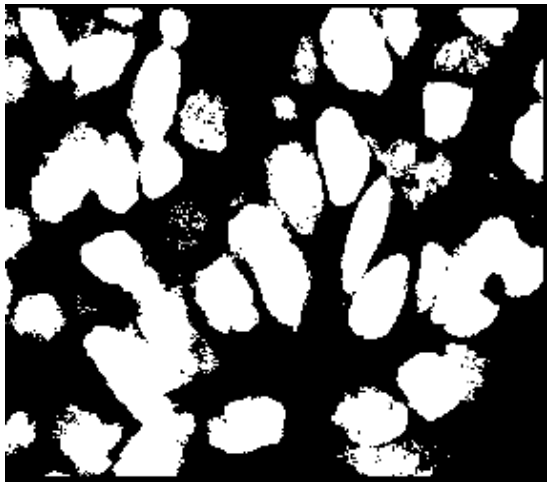


Slide credit: Bastian Leibe

เปรียบเทียบผลจากสองวิธี



ภาพอินพุต



Global threshold selection

(Otsu)

Local threshold selection

(Niblack)

Slide credit: Bastian Leibe

เรื่องที่ต้องการศึกษา

- ความรู้เบื้องต้นเกี่ยวกับภาพและการจัดการภาพ
- พื้นฐานการจัดการภาพในระดับพิกเซล
  - โมเดลสีพื้นฐานและการเปลี่ยนภาพสีให้เป็นภาพเฉดเทา
  - การทำเธรสโฮลด์แบบพื้นฐาน
  - การทำเธรสโฮลด์ด้วยวิธีของโอซึ (Otsu's Thresholding)
  - **Adaptive and Advanced Thresholding**
- พื้นฐานการเขียนโปรแกรมจัดการภาพ
  - การโหลดและแสดงรูป
  - การบันทึกภาพ
  - การแก้ไขค่าในรูปภาพ

ถ้ารู้งานที่จะเอาไปใช้ เรายังมีวิธีที่ลึกล้ำยิ่งกว่าอยู่อีก

Shijian Lu and Chew Lim Tan, Binarization of badly illuminated document images through shading estimation and compensation, ICDAR 2007.

- เจาะจงกับงานสแกนเอกสาร
- อาศัยข้อสังเกตที่ว่าภาพเอกสารมักจะมี smooth gradient (ความแตกต่างของเฉดสีแบบค่อยเป็นค่อยไป) ไม่ว่าแสงที่ฉายมาที่เอกสารจะสม่ำเสมอหรือไม่ก็ตาม
- หาค่าประมาณการให้แสงด้วยการทำ polynomial smoothing (Savitzky-Golay filter)

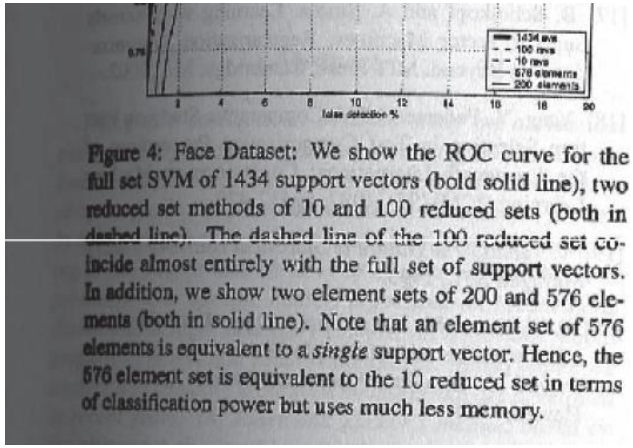
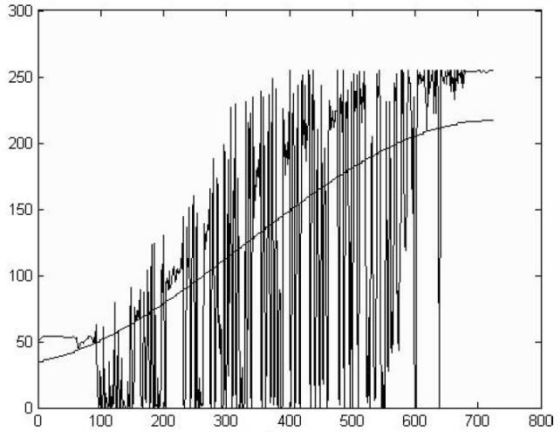
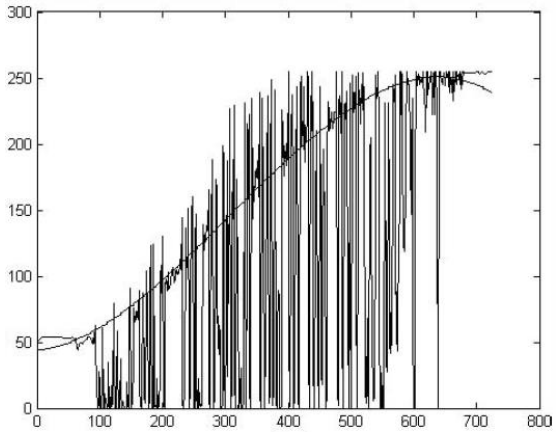


Figure 4: Face Dataset: We show the ROC curve for the full set SVM of 1434 support vectors (bold solid line), two reduced set methods of 10 and 100 reduced sets (both in dashed line). The dashed line of the 100 reduced set coincide almost entirely with the full set of support vectors. In addition, we show two element sets of 200 and 576 elements (both in solid line). Note that an element set of 576 elements is equivalent to a *single* support vector. Hence, the 576 element set is equivalent to the 10 reduced set in terms of classification power but uses much less memory.

อินพุต (เราจะพูดถึงข้อมูลที่เส้นสีขาวลากผ่าน)



Fit to the whole ‘white’ line



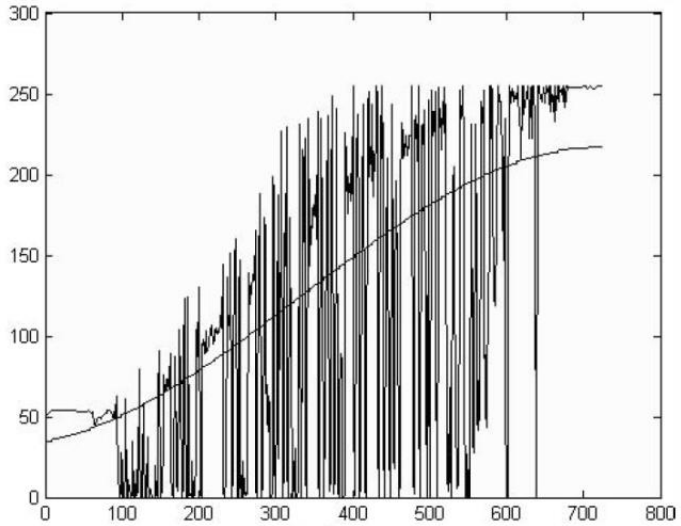
Fit to the background pixels

Source: Lu and Lim 2007

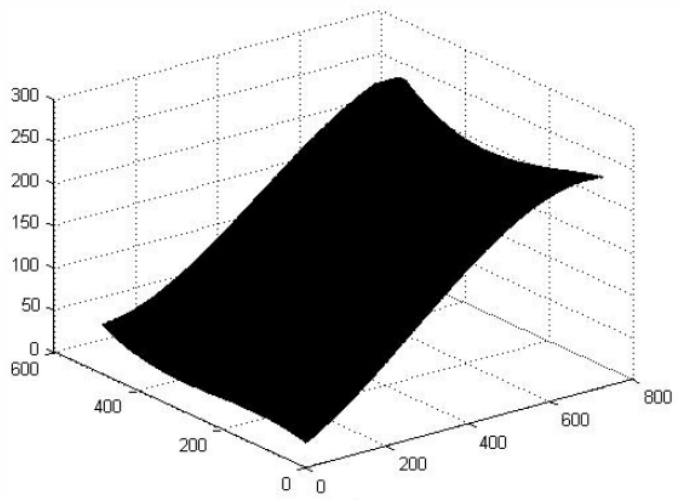
Lu and Tan's Method: Mathematic Viewpoint

- Polynomial Curve / Surface Fitting
  - ในไดอะแกรมที่แสดง เรานำข้อมูลเพียง scan line เดียวมาคิด แต่ที่จริงต้องทำทั้งเอกสาร
  - จากที่เราหา fitted polynomial curve เราต้องหา fitted polynomial surface แทน
  - Polynomial surface of degree *d*

$$z = f(x, y) = \sum_{i+j=0}^{d} b_{i,j}\, x^i y^j$$



Fitted polynomial line degree 3



Fitted polynomial surface degree 3

Source: Lu and Lim 2007

อธิบายสมการเพิ่มเติม

- ที่จริงแล้วสมการ

$$z = f(x,y) = \sum_{i+j=0}^{d} b_{i,j}\, x^i y^j$$

- ก็คล้ายกับสมการที่เราคุ้นเคย

$$y = a_3x^3 + a_2x^2 + a_1x + a_0$$

- ซึ่งจัดรูปใหม่ได้เป็น

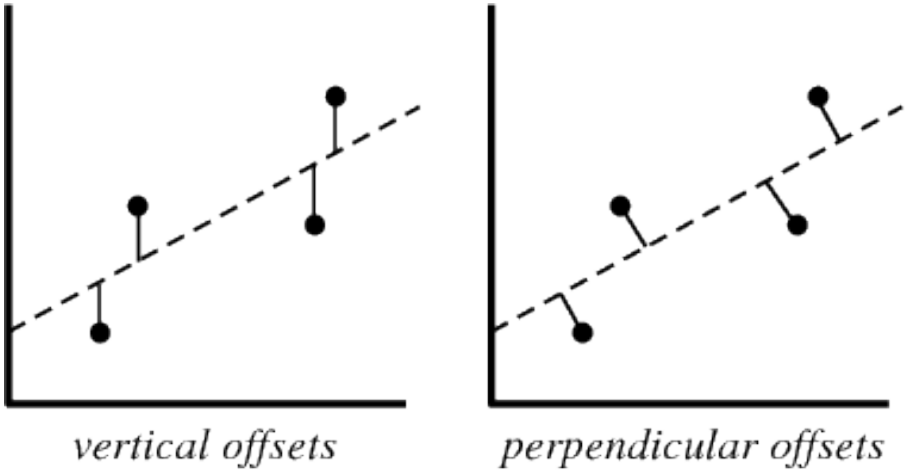
$$y = \sum_{i=0}^{3} a_i x^i$$

- ดังนั้นจากสมการแรก ถ้า d = 3 เราจะได้สมการใหม่เป็น $z = \sum_{i+j=0}^{3} b_{i,j} x^i y^j$

$$z = b_{3,0}\, x^3 + b_{2,1}\, x^2 y + b_{1,2}\, xy^2 + b_{0,3}\, y^3$$

ว่าแต่การ fit curve คืออะไร

- การหาเส้นโค้งที่ทำให้ผลรวมของระยะทางจากจุดที่สนใจมาถึงเส้นโค้งมีค่าน้อยที่สุด
  - เพื่อให้คำนวณได้โดยง่ายระยะทางที่ใช้จะเป็นระยะทางกำลังสอง (square distance)



  - ที่จริงระยะทางไปถึงเส้นควรจะเป็นระยทางแบบตั้งฉาก (perpendicular)
    แต่เพื่อทำให้การคำนวณสอดคล้องกับความเป็นตัวแปรอิสระและคำนวณง่ายขึ้นไปอีก

    ➔ ในทางปฏิบัติเราจะทำให้ผลรวมของ vertical offsets มีค่าน้อยที่สุด

- จากรูปแบบพื้นฐาน $y = b_1 x + b_0$ การ fit curve จะทำให้เราได้ค่า b_1 และ b_0

Source: Eric Weisstein

แล้วที่จริงเราทำการคำนวณอย่างไร

- ขั้นตอนการ fit curve / surface มีชื่ออย่างเป็นทางการว่า

Least-squares estimation

- คำนวณได้โดยการ minimize sum of least squares error

$$\mathbf{Ab} = \mathbf{v}$$

$$\begin{bmatrix} 1 & x_1 & y_1 & x_1^2 & x_1 y_1 & \cdots & y_1^3 \\ 1 & x_2 & y_2 & x_2^2 & x_2 y_2 & \cdots & y_2^3 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & y_n & x_n^2 & x_n y_n & \cdots & y_n^3 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

- จากสมการข้างต้น เรารู้ (x_i, y_i) ซึ่งเป็นตำแหน่งจุด และ v_i ซึ่งเป็นค่าเฉดเทา

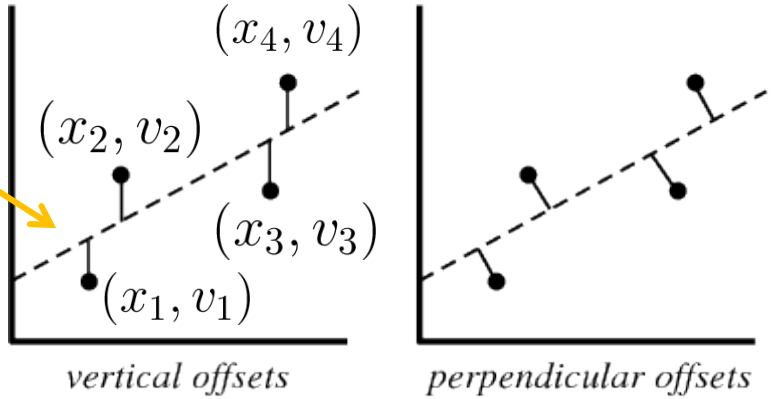
  ➔ แต่เรายังไม่รู้ b

แก้ A b = v ได้อย่างไร

$$\begin{bmatrix} 1 & x_1 & y_1 & x_1^2 & x_1y_1 & \cdots & y_1^3 \\ 1 & x_2 & y_2 & x_2^2 & x_2y_2 & \cdots & y_2^3 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & y_n & x_n^2 & x_ny_n & \cdots & y_n^3 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

- โดยปรกติแล้วไม่สามารถแก้ได้ **เพราะสมการไม่มีทางเป็นจริง!** (inconsistent equation)
- ต้องวกกลับมาที่พื้นฐานของการ fit curve ➔ ต้องการเส้นที่มี error น้อยสุด

$\hat{v} = \hat{b_1} + \hat{b_2}\, x$



เป้าหมายของเราคือจะหา $\hat{\mathbf{b}}$ ที่ทำให้

$$(\hat{v_1} - v_1)^2 + \ldots + (\hat{v_n} - v_n)^2$$

สูตรสำเร็จในการหา $\hat{\mathbf{b}}$

- มีทฤษฎีทางพีชคณิตเชิงเส้นรองรับไว้อยู่แล้ว เราสามารถหาผลเฉลย $\hat{\mathbf{b}}$ ได้จาก

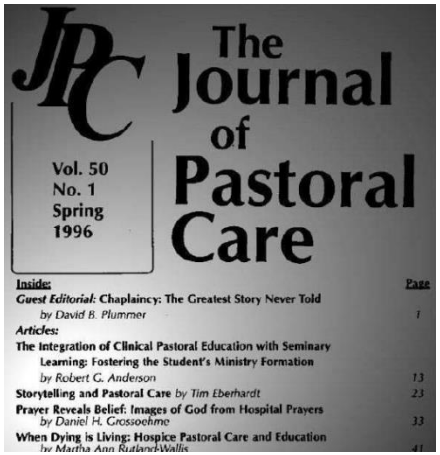
$$\hat{\mathbf{b}} = (A^T A)^{-1} A^T \mathbf{v}$$

- หรือใช้ pseudoinverse (Moore-Penrose inverse) ซึ่งมีรากฐานมาจากการทำ Singular Value Decomposition (SVD)

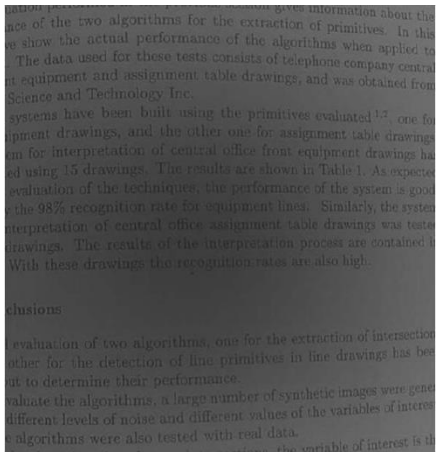
➔ ได้ผลออกมาเหมือนกัน และ Matlab มีคำสั่งที่เด็ดให้ใช้ทันที

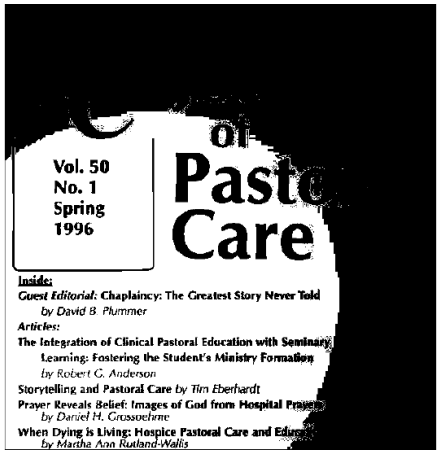
$$\hat{\mathbf{b}} = \hat{\mathbf{v}} \setminus A$$

ผลลัพธ์ดีขึ้นอย่างเห็นได้ชัด

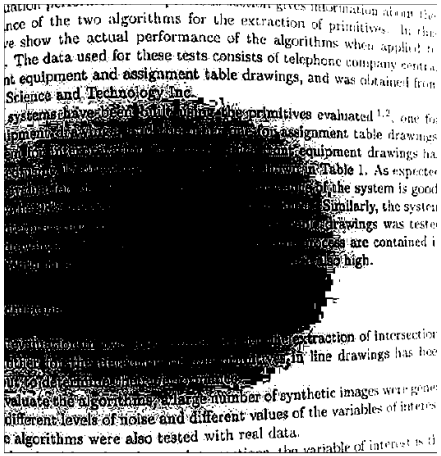


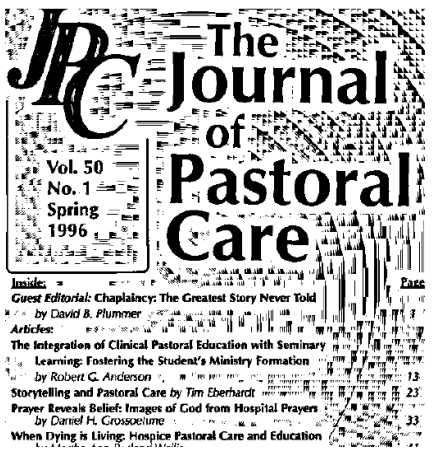
Input



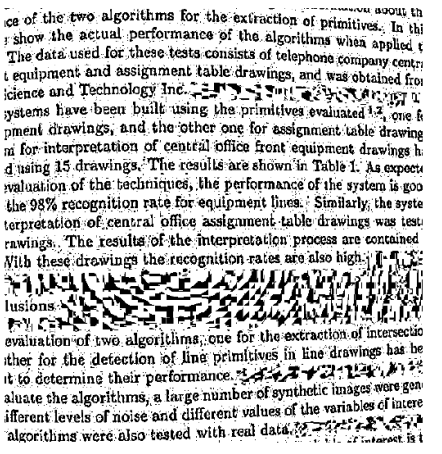


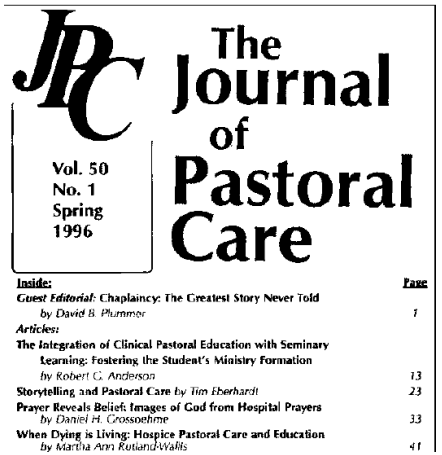
Otsu



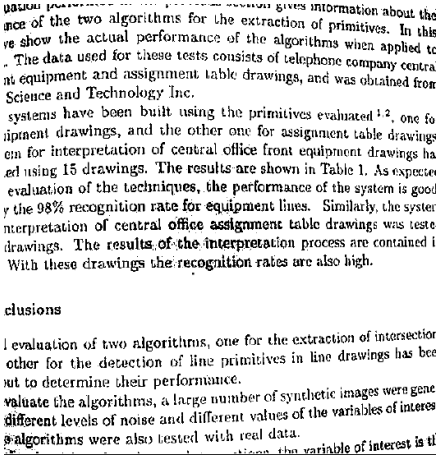


Niblack





Lu and Tan



ถึงวิธีการจะซับซ้อนกว่าของคนอื่น แต่ดูผลลัพธ์ก็รู้ได้ทันทีว่าอันไหนดีที่สุด → **คุ้มค่าที่จะเรียนพีชคณิตเชิงเส้น**

Source: Lu and Lim 2007

Lu and Tan เอา fitted surface ไปทำอะไร

- เอาไปใช้ปรับค่า threshold แบบวนซ้ำเป็นรอบ ๆ (iterative algorithm) ดังนี้

1. Fit parametric surface ด้วยจุดทั้งหมดในภาพ
2. ปรับค่าความสว่างด้วยการเอา fitted surface ไปลบค่าเฉดเทาในภาพ
3. ใช้ global threshold ด้วยวิธีของ Otsu
   → แยกพื้นหลังกับตัวอักษรได้ระดับหนึ่ง

แยกพื้นหลังขั้นต้น

4. Fit parametric surface ด้วยการใช้เฉพาะพิกเซลพื้นหลัง
5. ปรับค่าความสว่างด้วยการใช้ fitted surface จากขั้นตอนที่สี่
6. ใช้ global threshold ด้วยวิธีของ Otsu
   → แยกพื้นหลังกับตัวอักษรได้ดี

แยกพื้นหลังแบบละเอียด

7. ทำซ้ำ 4-6 ไปเรื่อย ๆ ถ้าจำเป็น

- วิธีทำงานได้ดีเพราะใช้สมมติฐานว่าพิกเซลส่วนใหญ่เป็นพื้นหลัง
- การแยกพื้นหลังขั้นต้นใช้พิกเซลตัวอักษรไปร่วมด้วย แต่แบบละเอียดจะใช้แต่พื้นหลังที่พบเท่านั้น

Slide credit: Bastian Leibe

เรื่องที่ต้องการศึกษา

- ความรู้เบื้องต้นเกี่ยวกับภาพและการจัดการภาพ
- พื้นฐานการจัดการภาพในระดับพิกเซล
  - โมเดลสีพื้นฐานและการเปลี่ยนภาพสีให้เป็นภาพเฉดเทา
  - การทำเธรสโฮลด์แบบพื้นฐาน
  - การทำเธรสโฮลด์ด้วยวิธีของโอซึ (Otsu's Thresholding)
  - Adaptive and Advanced Thresholding
- **พื้นฐานการเขียนโปรแกรมจัดการภาพ**
  - **การโหลดและแสดงรูป**
  - การบันทึกภาพ
  - การแก้ไขค่าในรูปภาพ