Computer Science 1001.py

Lecture 26[†]: Visual Secret Sharing Concluding Remarks

Instructor: Benny Chor

Teaching Assistant (and Python Guru): Rani Hod

School of Computer Science
Tel-Aviv University
Fall Semester, 2011/12
http://tau-cs1001-py.wikidot.com

Lecture 25: Topics

- Global and local alignment.
- n-out-of-n secret sharing.
- Visual secret sharing.

Lecture 26: Plan

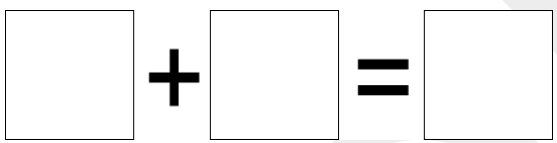
- Visual secret sharing.
- Concluding remarks.

And Now, to the Last Topic in our Course : Secret Sharing and Visual Secret Sharing

Visual Secret Sharing (Moni Naor and Adi Shamir, 1994)

Instead of shares and secrets being numbers or bit sequences, we now consider a scenario where both the shares and the secrets are images. Each individual secret should still give no information about the secret. To reconstruct a secret from the shares, we simply overlay the two images one on top of the other. Can this be done?

An initial examination will indicate that this idea makes little sense. In the physical-optical world, a black pixel blocks light through it, and this is irreversible. Pixel optics "implements" bitwise `or`, while secret sharing is using bitwise `xor`:



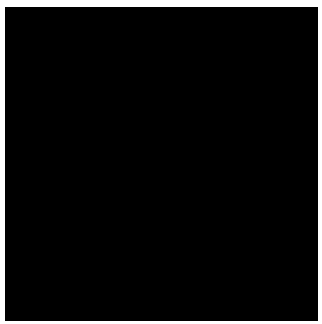
$0 + 0 = 0$

$1 + 0 = 1$

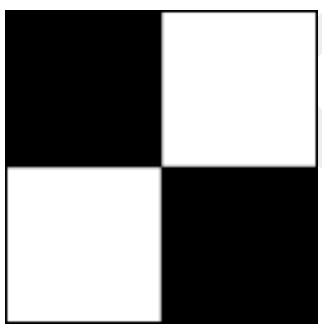
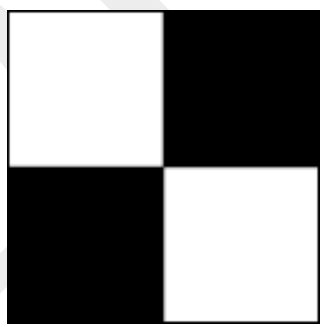
$1 + 1 = 1$

Doubling the Pixel (or, Thinking Outside the Box)

Who said one bit ought to be represented by a single black/white pixel? Lets go to two-by-two pixels to represent (just) one bit. The bit 1 in the secret will be represented by a solid black: All four pixels being black:

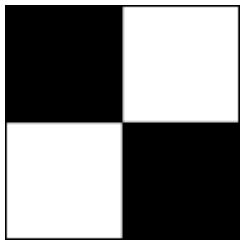
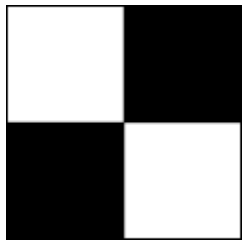


The bit 0 in the secret will be represented by a two diagonal black out of the four pixels. So there are two representations of the bit 0:

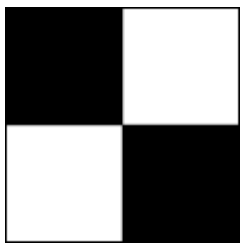
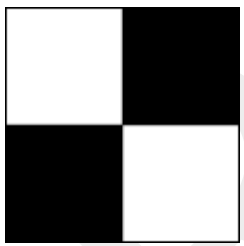
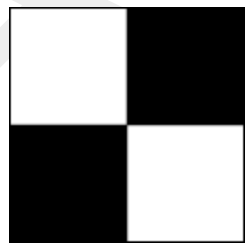
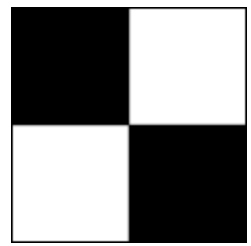
 and  .

Generating Shares

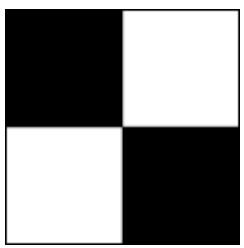
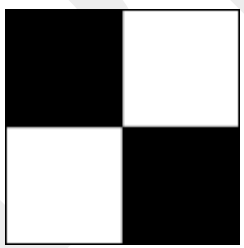
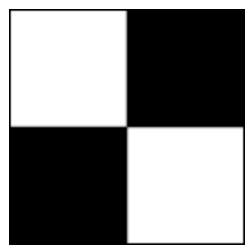
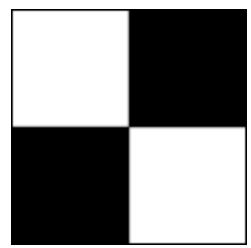
Each share of one bit is, with probability exactly $1/2$, either

 or  .

If the secret bit is 1, the two shares are either (with prob. $1/2$ each):

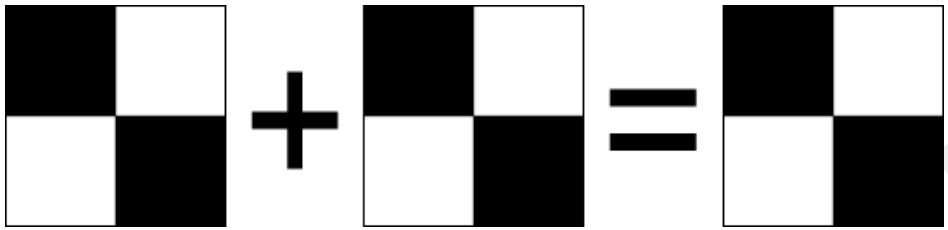
 and  , or  and  .

If the secret bit is 0, the two shares are either (with prob. $1/2$ each):

 and  , or  and  .

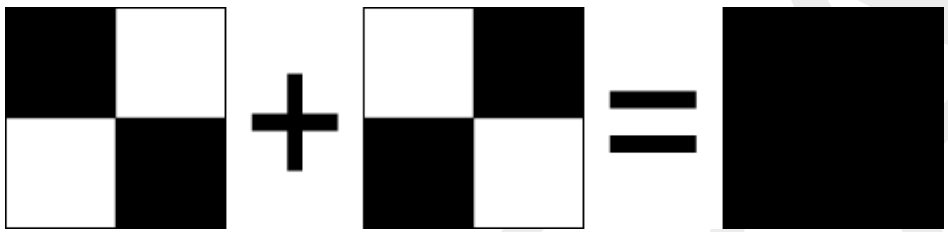
Combining Shares

If the secret bit is 0, we have either



or the dual configuration.

If the secret bit is 1, we have either

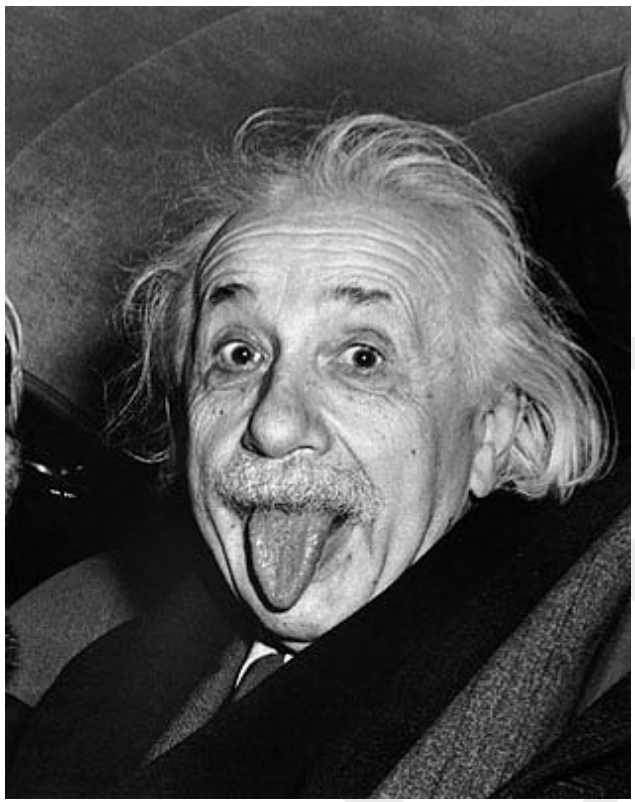


or the dual configuration.

Any black and white image can be shared among two parties, bit by bit. The shares carry no information about the secret or any part thereof. If the original image is `n-by-m` pixels, the two shares and the reconstruction will be `2n-by-2m` pixels. The visual quality of the reconstruction will not be as good as the original (but it can be processed to reproduce the original black-and-white image).

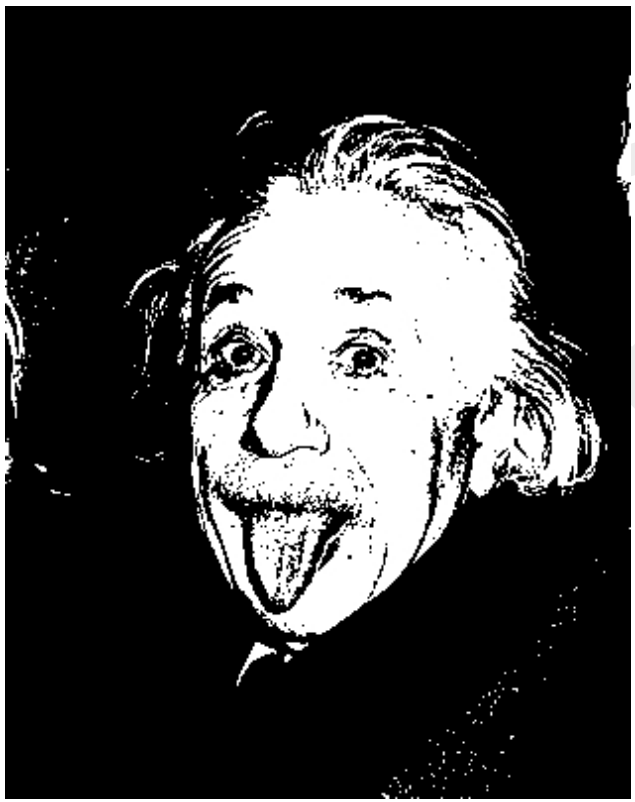
A Concrete Example

We start with a familiar image



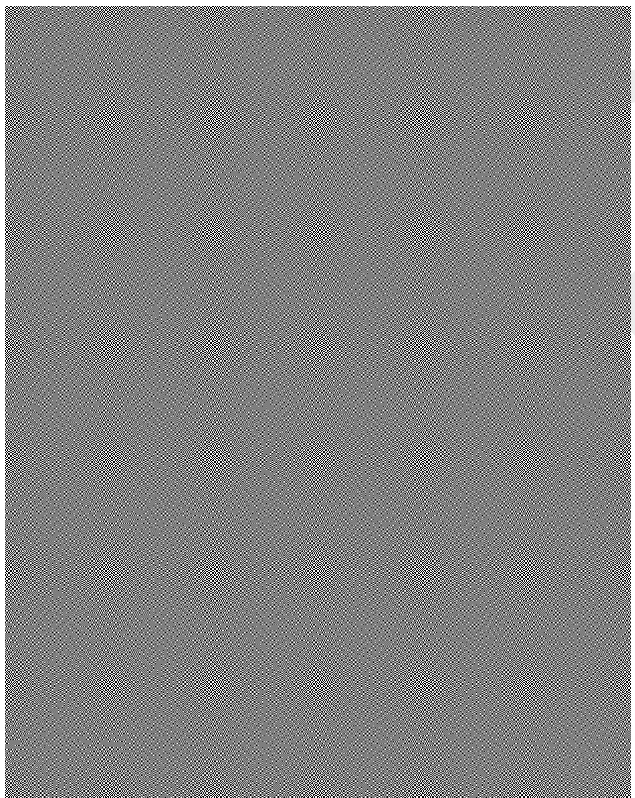
Stage 1: Discretization

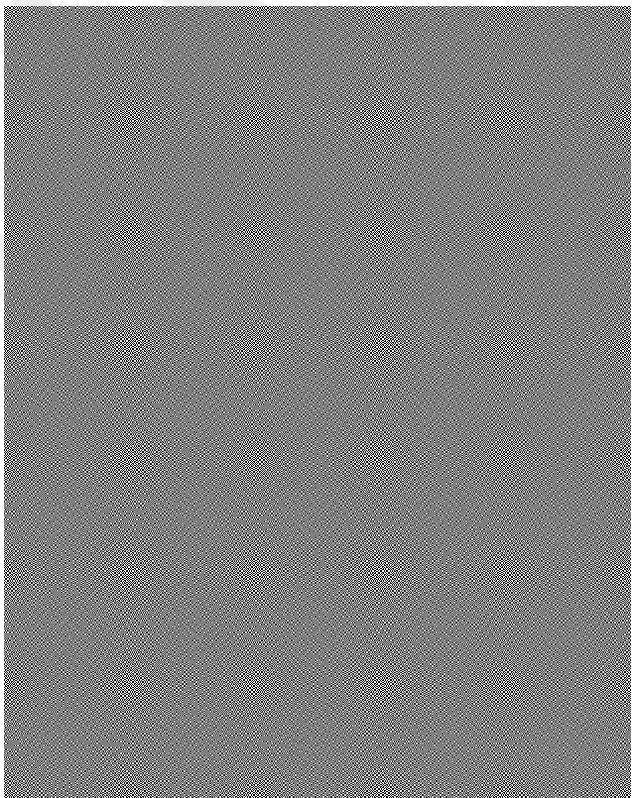
The original grey level image is discretized: We construct a new image, with the same shape as the original. For each pixel with coordinates `[i,j]`, its value is compared to a threshold (default value 128). If the value is greater or equal to the threshold, we set the `[i,j]` pixel in the new image to `255` (white). Otherwise, the value is set to `0` (black). The visual quality of the image is reduced.



Stage 2: Randomize and Create Shares

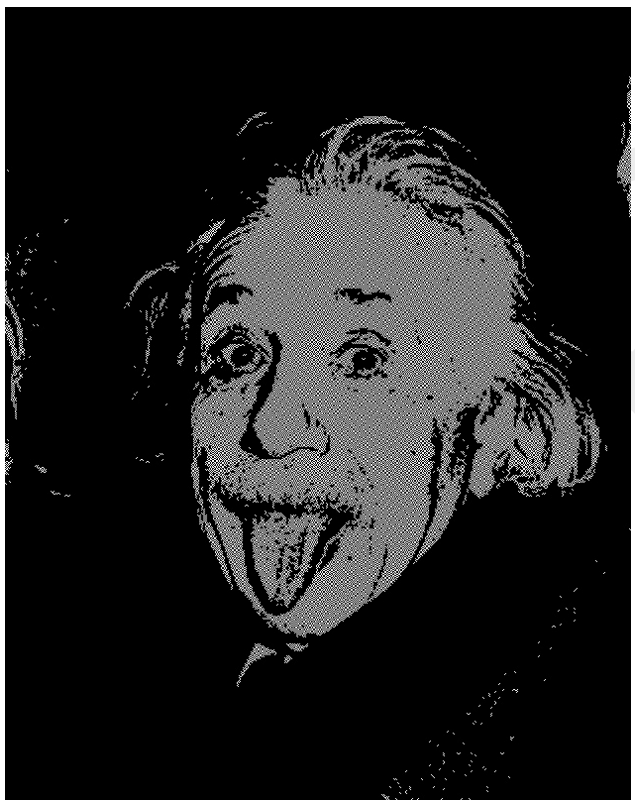
We construct two new image (shares), each with double the shapes as the original. For each pixel with coordinates `[i,j]`, in the original, we create four pixels with coordinates `[2i,2j]`, `[2i+1,2j]`, `[2i,2j+1]`, `[2i+1,2j+1]` in the shares. Out of these, exactly two are white and exactly two are black in each share. Their mutual configuration depends on the value of the original `[i,j]` pixel, as explained earlier, and on the outcome of the `[i,j]` coin toss.





Stage 3: Reconstructing the Secret from the Shares

Given the two equal shape (shares), we reconstruct the secret, of the same shape. Each pixel with coordinates `[i,j]`, in the secret, will attain the minimum of the two values in the same coordinates `[i,j]` in the secrets. Essentially, this means that if one of the shares is black in this coordinate, so will be the secret. Otherwise, this pixel in the secret will be white.



Stage 1, Discretization: Python Code

Discretize is done deterministically. Each pixel value is compared to a threshold (default value is 128=grey), and set to black or white accordingly.

```
import numpy
from PIL import Image
import random

def discretize(pic,threshold=128):
    """ discretizes each pixel into 4 adjacent pixels of black (0)
    and white (255). Uses a threshold whose default value is grey "
    n,m=pic.shape
    A=numpy.zeros((n,m),dtype='int8')+255

    for i in range (n):
        for j in range (m):
            if pic[i,j]>threshold:
              new=255
            else:
              new=0
            A[i,j]=new
    return A
```

Stage 2, Randomize and Create Shares: Python Code

```
def split(A):
    """ split A at random to two correlated matrices B,C.
    For each 2-by-2 "proper" submatrix of both B,C, either
    the two pixels on the diagonal are 0 and the anti diagonal
    255, or the opposite (this is determined by a coin flip)"""

    n,m=A.shape
    B=numpy.zeros((2*n,2*m),dtype='int8')+255
    C=numpy.zeros((2*n,2*m),dtype='int8')+255

    white=["*","*"]
    white[0]=numpy.array([[1,0],[0,1]])*255
    white[1]=numpy.array([[0,1],[1,0]])*255

    for i in range (n):
        for j in range (m):
            coin=random.randint(0,1)
            if A[i,j]==255:
                B[2*i:2*i+2,2*j:2*j+2]=white[coin]
                C[2*i:2*i+2,2*j:2*j+2]=white[coin]
            elif A[i,j]==0:
                B[2*i:2*i+2,2*j:2*j+2]=white[coin]
                C[2*i:2*i+2,2*j:2*j+2]=white[1-coin]
    return B,C
```

Stage 3, Reconstructing the Secret from the Shares: Python Code

This is the final and simplest step. Each pixel `[i,j]` in the two shares is examined. The output is the `minimum` of the two. This means that if one of them is black, the pixel `[i,j]` in the result is black. If the two are white, so is the result. In `numpy` there is an explicit command for taking the pixel wise minimum of two arrays/matrices.

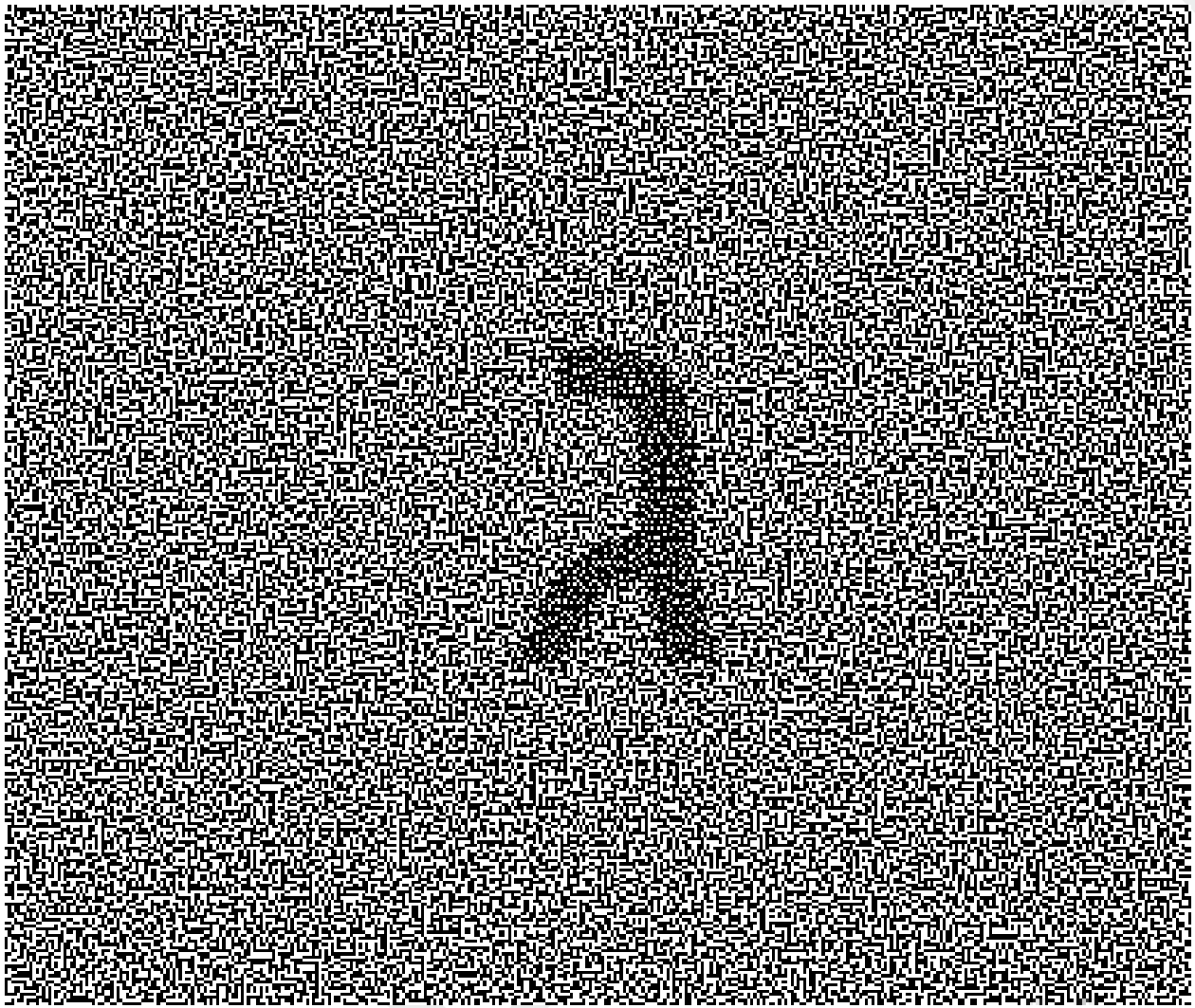
```
def combine(share1,share2):
    if share1.shape != share2.shape:
        print("pictures of incompatible sizes")
        return None
    else:
     return numpy.minimum(share1,share2)
```

All that is left to do now is to convert the three matrices back to images.

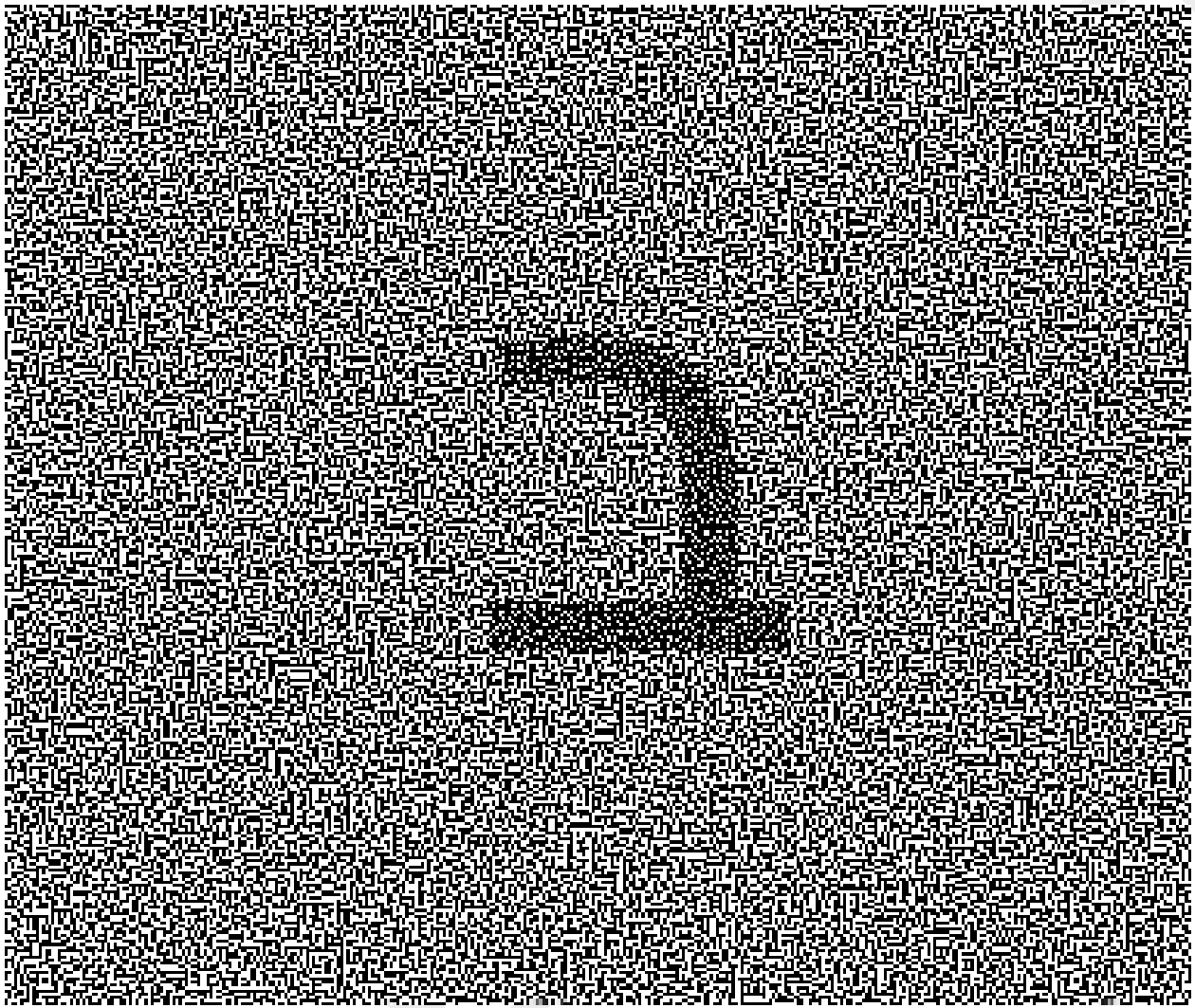
An Even Fancier Visual Secret Sharing Scheme

We are given three images (of equal shapes), `target`, `origin1`, `origin2`. We generate three new images `secret`, `share1`, `share2`, where `share1` is a transformation that looks like `origin1`, `share2` is a transformation that looks like `origin2`, and their combination (putting one on top of the other) yields an image, `secret`, that looks like `target`.

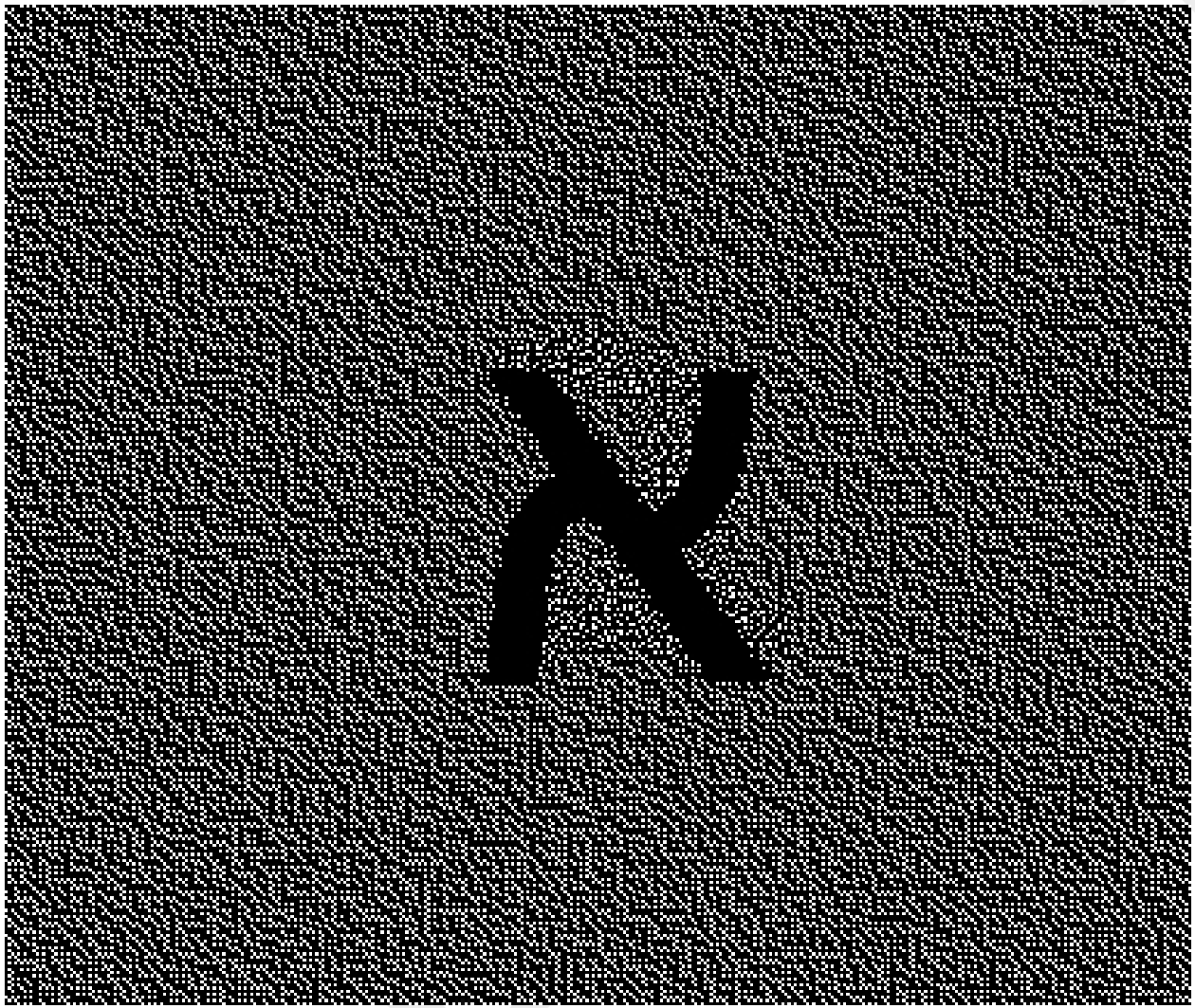
A Concrete Example: Share1



A Concrete Example: Share2

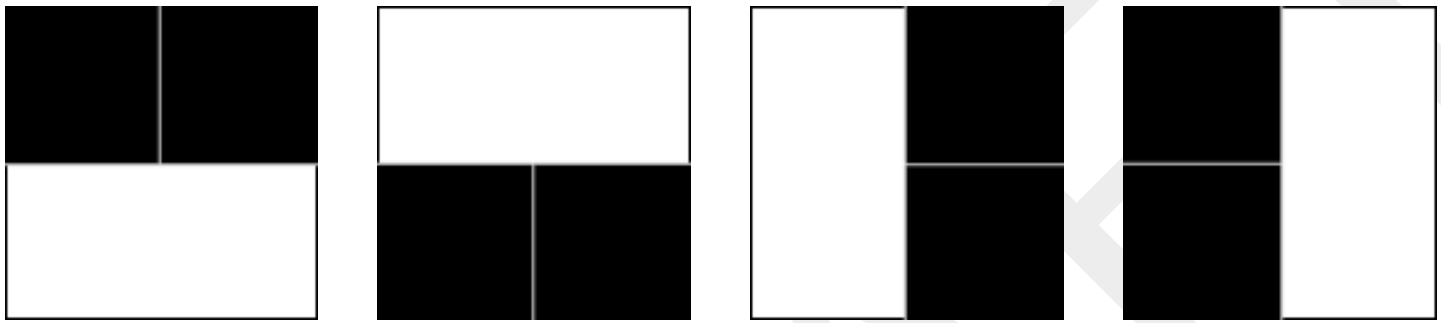


A Concrete Example: Target (Secret)



The Fancier Scheme: A Few Details

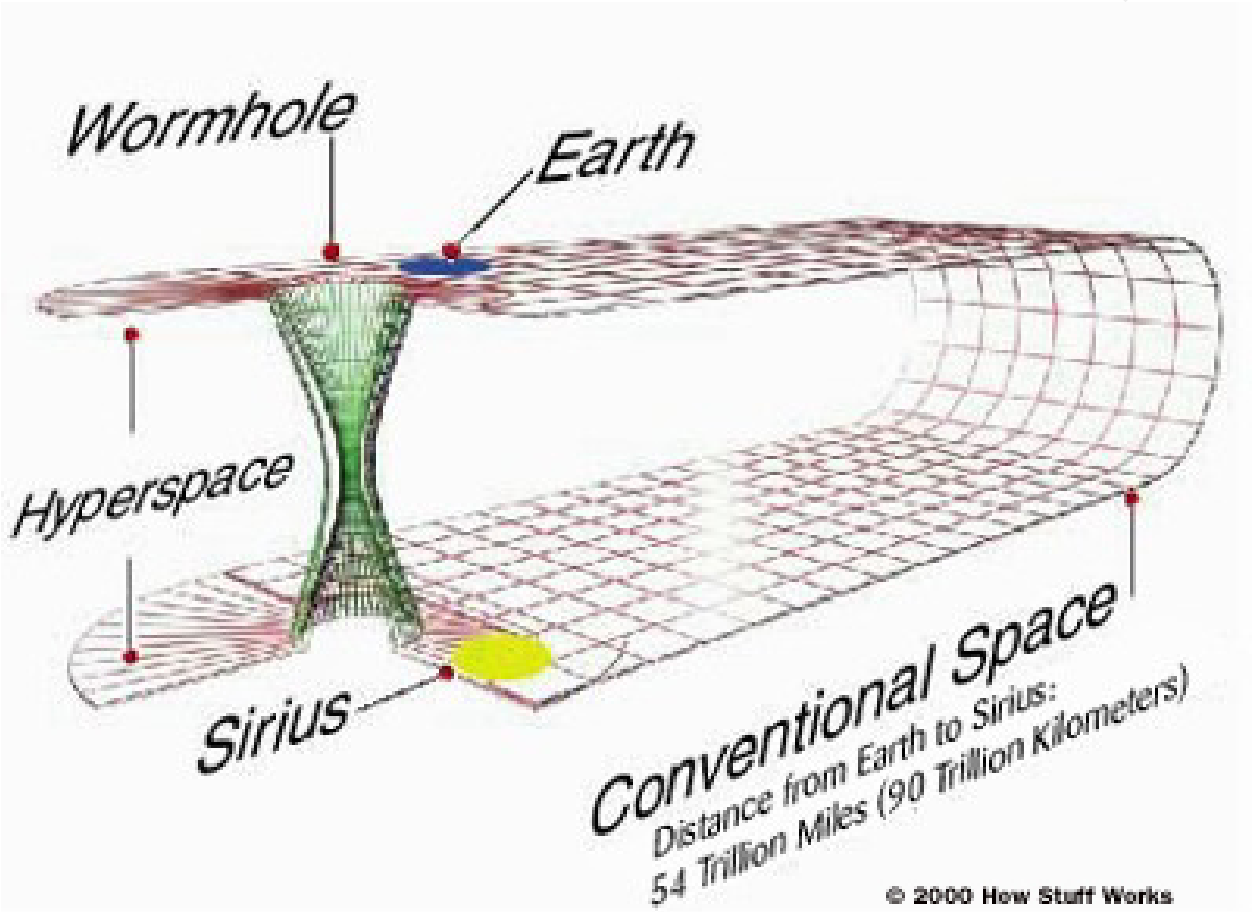
For the shares, an original white pixel is encoded by one of four alternatives:



For the shares, an original black pixel is encoded by one of four alternatives:

Their combinations depend on the target pixel. For the secret, a black is a solid 2-by-2 blacks, while a white is one of the 3 blacks out of 4 configurations above. We omit further details.

And Now to Something Completely Different: A Brief Voyage Back in Time



Let us go back to Sunday, Oct. 30, 2011.

Travel Advisory

- You are about to take a new version of the "extended introduction to CS" course.
- This is the second run ever of this course.
- About half of first year CS students this year take this version.
- The older version of the course is still up and running.
- The programming language used in the older version is Scheme.
- The programming language used in the new version is Python.
- It is your choice (using the bidding system) which version to take (both have advantages and disadvantages).
- There are many good reasons to take the older version of the course.

Travel Advisory (cont.)

- There are many good reasons to take the older version of the course.
- The older version is much more stable, better documented and better supported than the new version.
- A large number of exams and homework assignments for the older version are available online.
- The older version has a universally acclaimed text book.
- The new version has none (acclaimed or not).
- The lecturer of the new version is erratic and unpredictable.
- But the teaching assistant more than makes up for this :=)

- Welcome aboard!

Administrative Details

- Intended for 1st year Computer Science students.
- Grade determined by exam (70-80%) and homework (30-20%).
- In order to pass the course, you must pass the exam and get a passing grade in at least $n-2$ out of the n home assignments.
- Homework grade is best $n-2$ out of n assignments.
- Exam on March 2nd, 2012 (Moed B on Marth 30th).
- Exam is closed book except for 2 double sided, normal size (A4) pages.

Administrative Details (2)

- ~~7–9~~ 6 assignments, each with both "dry" and "wet" component.
- "Wet" homework submission in groups of size one or two (but not three or more).
- "Dry" parts of homework submitted in groups of size exactly one.
- Submissions after the deadlines will not be considered unless coordinated in advance (*e.g.* in case of reserve service) or reported asap and accompanied by valid documention (*e.g.* in case of serious illness).
- If one member of a pair has a valid reason for late submission, the other member is still expected to meet the deadline on his/her own.
- Office hours (both Benny & Rani): By e-appointment.
- E-mails: benny AT cs.tau.ac.il , ranihod AT tau.ac.il
- Course site: http://tau-cs1001-py.wikidot.com ~~(not operational yet)~~.

Course Structure

- We will "sample" 10–14 different topics of interest from a fairly wide range in Computer Science.
- This should hopefully expose you to some of the beautiful topics and ideas in the field.
- Naturally we will not get into any of them in great depth.
- We leave this to the required and elective courses you will take later in your studies.
- Each topic will be accompanied by programming examples (given by us) and tasks (to be done by you).
- Lectures and recitations are coordinated and both form integral parts of the course.

Programming Aspects

- This is not a programming course.
- Yet, you will learn and use a specific programming language – Python.
- And basic ideas in programming like iteration, control structure, recursion, procedures, objects, basic data structures, etc. etc.
- Python is a relatively new programming language, gaining popularity in many applications.

- Those of you who heard about Scheme and were hoping to learn it here, should take the other intro CS class.

One More Thing this Course is Not About

Course Topics
(tentative list, not in order, somewhat ambitious)

- Python programming basics (2–3 meetings)
- Bits, bytes, and representation of numbers in the computer.
- Huge integers, with applications to public key cryptography.
- Representing and manipulating images.
- Text compression.
- Simple error correction codes.
- String matching.
- Hashing and hash functions.
- Comparing and aligning sequences, using dynamic programming.
- Numerical computations (Newton–Raphson root finding).
- Spatial pattern identifications via geometric hashing.
- Hard computational problems, and approaches to solving them.
- Secret sharing.

Course Topics

(What we did cover – not a complete list)

- Python programming basics (2–3 meetings)
- Bits, bytes, and representation of numbers in the computer.
- Huge integers, with applications to primality testing and public key cryptography. GCD.
- Representing and manipulating images.
- Text compression (Huffman, Codebook, Ziv-Lempel).
- Simple error correction codes (Hamming, Hadamard, others).
- String matching.
- Hashing and hash functions.
- Comparing and aligning sequences, with applications in computational biology and in computer music.
- Numerical computations (Newton–Raphson root finding).
- ~~Spatial pattern identifications via geometric hashing~~.
- ~~Hard computational problems, and approaches to solving them~~.
- Secret sharing and visual secret sharing.
- The infrastructure of the net (guest lecture by Eyal Dagan).

More Course Topics (What we did cover – not a complete list either)

- Lists slicing and list comprehension. Immutable and mutable objects.
- Recursion and memoization. Higher order functions.
- Defining functions using `lambda` expressions.
- Computational complexity: Basic notions. Big O notation. Binary vs. unary representation.
- Sets and dictionaries; Iterators and generators.
- Object oriented programming (classes and methods).
- Random number generation; Timing operations; File I/O.
- Numerical stability (in the context of Newton-Raphson and of matrix inversion).
- Faster matrix multiplication (Strassen).
- Mandelbrot's fractal.
- The hare and turtle method. Application to integer factoring.

Plan for the Rest of the Course

- A few words about the dreaded exam
- Some propaganda (two short advertisements).
- Thanks.
- Surprise.
- Famous last words.

The Dreaded Exam

- All material covered in class, recitations, and homework, from all parts of course.
- Both multiple choice ("closed") and "open" questions.
- Including some coding ("dry coding") and/or fixing bugs in given code.
- Make sure you verbally explain your code.
- You can bring and use two double sided A4 (normal size) pages.
- If we deem some details are required, we'll supply them.
- Do not waste effort and memory trying to memorize obscure details.
- But knowledge of basic ones (*e.g.* slicing, list comprehension) is expected.

- Piece of cake.

Short Advertisement (1):
Teaching Computer Science in the community
(mostly for 3rd year students)

Your chance to spread your knowledge to school kids (typically mid school, ages 12-15). A challenging and personally rewarding activity.

Short Advertisement (2): Student exchange (2nd year +)

סמסטר בוונציה

אוניברסיטת תל-אביב

תלמידים לתואר הראשון באוניברסיטת תל-אביב מוזמנים ללמוד
במשך סמסטר אחד באוניברסיטה הבינלאומית של ונציה
Venice International University (VIU) www.univiu.org

בקורסים בתחומים הבאים
(שפת ההוראה – אנגלית):

היסטוריה, תיאוריה של האמנות, תולדות האמנות, תקשורת,
אדריכלות, לימודי דתות, תורת הספרות, סוציולוגיה,
מדע המדינה, כלכלה, איטלקית מדוברת

ההרשמה לסמסטר א' תשע"ב היא עד 30/04/2011
ולסמסטר ב' תשע"ב - עד 15/11/2011

מידע על הכרה בקורסים של VIU יינתן על-ידי החוגים / הפקולטות
למספר תלמידים מצטיינים שהרשמתם תאושר יוענקו מלגות בסך 800$ כל אחת

לגבי פרטים על ההרשמה, שכר הלימוד, המגורים ומידע נוסף נא לפנות אל
אבי זרכובסקי, המזכירות האקדמית, בניין הסנאט ע"ש ג'ורג' ס' וייז, חדר 202 בימים א'-הי,
בין השעות 10:00-13:00, טלפון: 6409989, דואל: zaravi@post.tau.ac.il

VIU is just one attractive example out of many options. For a full list of universities with whom TAU has a student exchange agreement, look at the site of TAU office of international academic affairs, or send e-mail to acadaff@tauex.tau.ac.il.

Many Thanks

To the people who taught me about topics I was not familiar with and gave me feedback and advice about earlier versions of the course material and slides (too many to list individually).

To the students in the course who contributed pieces of code, error reports, and constructive criticism.

And especially to Rani Hod, for being such a professional, knowledgeable, and cool TA.

Famous Last Words

You have brains in your head.
You have feet in your shoes.
You can steer yourself
any direction you choose.
You're on your own. And you know what you know.
And YOU are the guy who'll decide where to go.

By Theodor Seuss Geisel (aka Dr. Suess, 1904–1991).
Hebrew translation by Leah Naor.

For an alternative version of the book Oh, the places you'll go, by Dr. Seuss, as told[‡] by the people of Burning Man 2011, turn to youtube.

[‡] in English only. The Gujarati version was, ahmm, lost in translation.