模式识别

HMM: Hidden Markov Model – part 2

隐马尔科夫模型第二部分

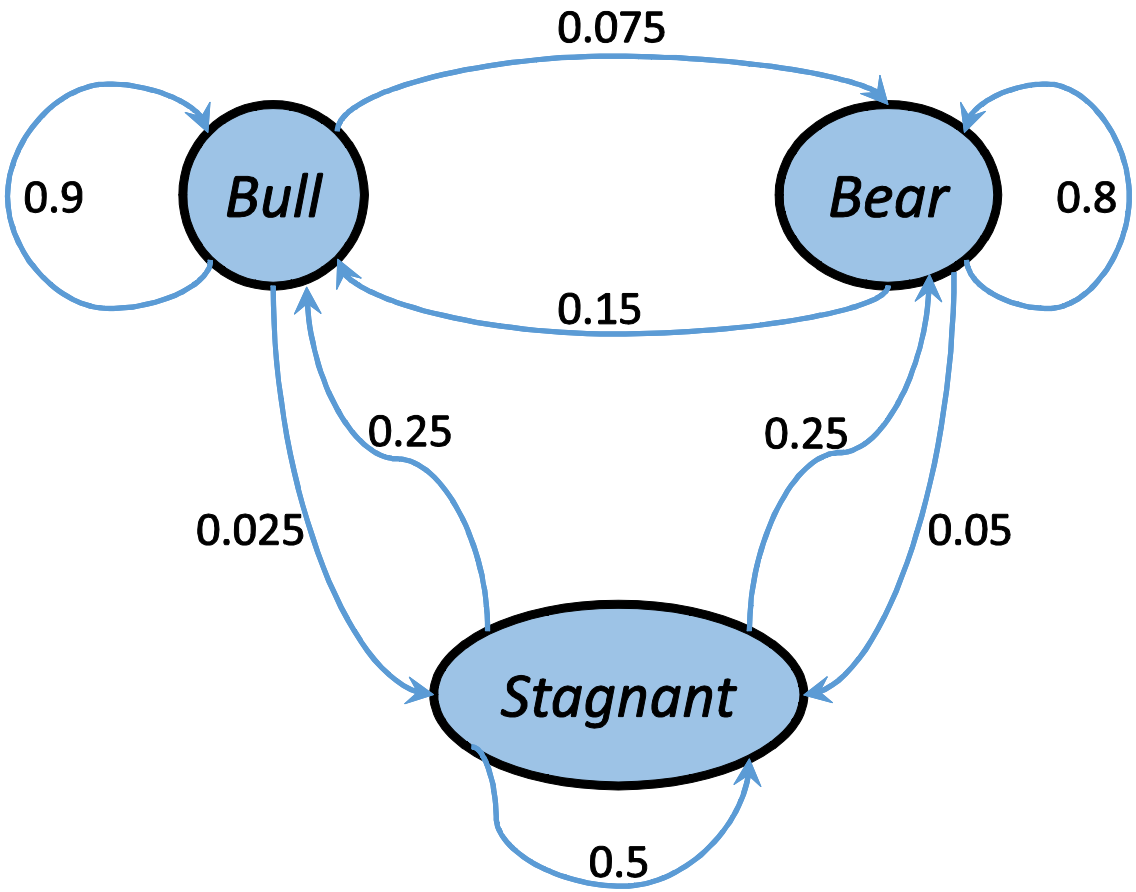
张振宇

南京大学智能科学与技术学院

2025

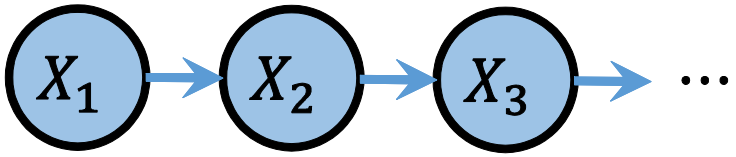
Markov Chain马尔科夫链

- ✓ Markov chain (discrete-time Markov chain or DTMC)



可视化和形式化

- ✓ 可视化：



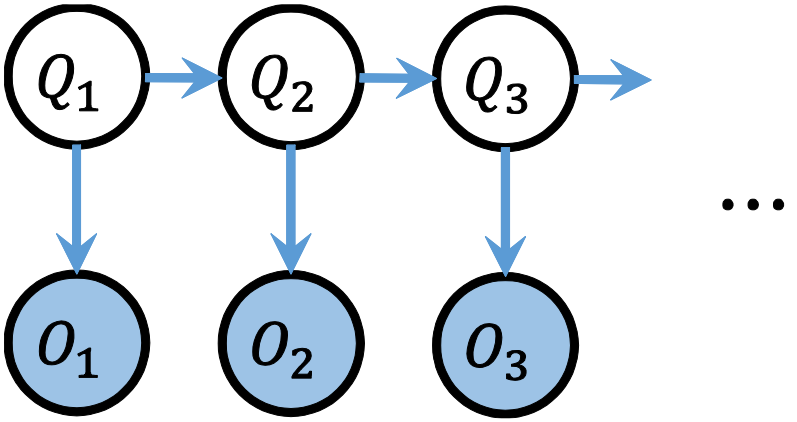
  - 注意填充的变量表示观察值（即随机变量值已知）
- ✓ 那么，如何形式化定义DTMC？需要哪些量？
  - 系统初始化Initialization: $P(X_1)$或者$X_1 = x_1$
  - Transition probability: $A = P(X_{t+1}|X_t)$
  - 还需要别的吗？
  - 两次运行结果会一样吗？

转移概率矩阵

- ✓ Transition probability matrix转移概率矩阵
  - A是一个$N \times N$的矩阵
  - $A_{ij} = P(X_t = j | X_{t-1} = i)$
  - 行和为1！
- ✓ 如果运行足够久（$t \to \infty$），那么X_t的分布在很多情况下将稳定下来，叫Stationary distribution，记为$\boldsymbol{\pi}$
  - $\boldsymbol{\pi} = A\boldsymbol{\pi}$
- ✓ Google PageRank的简化!

怎样在模式识别中发挥更大作用？

- ✓ 例如，在连续手写识别中，笔画stroke没有在DTMC里面用到，那么DTMC就没法用于连续手写识别
  - 想办法把笔画加进去！
  - 状态是什么？



形式化

- ✓ Q：隐变量(hidden variable)，不可观测的状态
- ✓ N： number of states 状态数,N个可能的状态为$\{S_1, \dots, S_N\}$
- ✓ $O(o)$：观察值(observations)，M个可能的观察值 $\{V_1, V_2, \dots, V_M\}$
- ✓ T：时间序列的长度
- ✓ $\boldsymbol{\pi}$：初始化, $\pi_j = P(Q_1 = S_j)$
- ✓ A： transition probability matrix, $A_{ij} = P(Q_{t+1} = S_j | Q_t = S_i)$
- ✓ B： emission probability 发出观察值的概率（发射概率）
  - $b_j(k) = \Pr(O_t = V_k | Q_t = S_j)$
  - 假设B不随时间变化，当未知状态为j时观察到为k的概率
  - 那么，j，k的取值范围是？B的行和是？

HMM中要解决的问题

- ✓ 怎样设计状态？-- 自动学习？
- ✓ 怎样设计观察值？--根据问题的特点和实践反复设计

- ✓ 与具体问题无关的
  - 指定一个HMM需要的所有参数：$\lambda = (\boldsymbol{\pi}, A, B)$
  - 问题1：Evaluation估值
  - 问题2：Decoding解码
  - 问题3：Learning学习

Problem 1. Evaluation

- ✓ 输入
  - 一个完全指定的HMM模型，即$\lambda = (\boldsymbol{\pi}, A, B)$已知
  - 一个完全观测的输出序列$O_1 O_2 \cdots O_T$，或$\boldsymbol{O} = O_{1:T}$
- ✓ 输出
  - $P(\boldsymbol{O}|\lambda)$ – 含义是？
  - 在这个模型λ中观察到特定输出$\boldsymbol{O}$的概率
- ✓ 作用是？
  - 可以看出此模型对该观察序列的成绩score
  - 可以用来从多个模型中选择最适合的模型

Problem 2： Decoding

- ✓ 输入
  - 一个完全指定的HMM模型，即$\lambda = (\boldsymbol{\pi}, A, B)$已知
  - 一个完全观测的输出序列$O_1 O_2 \cdots O_T$，或$\boldsymbol{O} = O_{1:T}$
  - 某个标准criterion
- ✓ 输出
  - 一个完全指定的隐变量序列$X_{1:T}$的值
- ✓ 作用是？
  - 如，语音识别中状态可能有实际意义（各音节）
    - 唯一吗？
  - 可以用来观察模型结构，优化模型

Problem 3： Learning

- ✓ 输入
  - 网络结构，状态数、输出数
  - 若干观测序列$\{\boldsymbol{O}\}$
- ✓ 输出
  - 最优的参数$\lambda = (\boldsymbol{\pi}, A, B)$使得$P(\{\boldsymbol{O}\}|\lambda)$最大
- ✓ 作用
  - 显而易见
  - 最重要的问题
  - 有时候一个足够长的观测序列就够了

Evaluation

假设隐状态已知

- ✓ 已知$\lambda, o_{1:T}$，求$P(o_{1:T}|\lambda)$
- ✓ 若假设oracle已告知所有的隐变量的值$q_{1:T}$
  - $\Pr(o_{1:T}|\lambda, q_{1:T}) = \prod_{i=1}^{T} \Pr(o_t|q_t, \lambda) = \prod_{i=1}^{T} b_{q_i}(o_i)$
  - 证明？含义？
  - λ的存在只是表明概率的大小是基于该模型参数计算的，可以去除而不影响计算
- ✓ 关于各随机变量之间的独立性的判断，进一步参阅PRML第八章

一种naïve的计算方法

- ✓ 那么隐变量序列$q_{1:T}$的可能性多大呢？
  - $\Pr(q_{1:T}|\lambda) = \pi_{q_1} A_{q_1 q_2} A_{q_2 q_3} \cdots A_{q_{T-1} q_T}$
  - 含义？
- ✓ 用全概率公式对所有可能的$q_{1:T}$求和可以得到$\Pr(o_{1:T}|\lambda)$
  - $\Pr(o_{1:T}|\lambda) = \sum_{all\ Q} \Pr(o_{1:T}|\lambda, q_{1:T}) \Pr(q_{1:T}|\lambda)$ ，复杂度？
  - $O(T \times N^T)$
- ✓ 虽然不实用，但可以从中学到一种思考问题的方法

那么，如何快速计算？

- ✓ 动态规划！$\Pr(o_{1:T}|\lambda)$可以拆分成$\Pr(o_{1:T-1}|\lambda)$和$\Pr(o_T|\lambda)$

- ✓ 只看最后一步（$t = T$)，该如何计算?
  1. 最后一步（$t = T$)时一共可能有N种状态：$q_T = S_1, \ldots, S_N$，其概率$\Pr(o_{1:T-1}, Q_T = S_i\,|\lambda) = ?$
  2. 若最后一步状态为S_i, 那么观察到输出o_T的概率是多少？
  3. 所求的值是多少？(全概率公式)

$$\Pr(o_{1:T}|\lambda) = \sum_{i=1}^{N} \Pr(o_{1:T-1}, Q_T = S_i|\lambda)b_i(o_T)$$

  - 只限于最后一步吗？

快速计算（2）

✓ 如何计算 $\Pr(o_{1:T-1}, Q_T = S_i|\lambda)$？

- 有N种可能，即$T-1$时刻状态为$q_{T-1} = S_j$，$j = 1,2,\dots,N$，然后通过概率A_{ji}转移

- 全概率公式，again！

$$\Pr(o_{1:T-1}, Q_T = S_i|\lambda) = \sum_{j=1}^{N} \Pr(o_{1:T-1}, Q_{T-1} = S_j|\lambda) A_{ji}$$

快速计算小结

- ✓ $\Pr(o_{1:T}|\lambda) = \sum_{i=1}^{N} \Pr(o_{1:T-1}, Q_T = S_i|\lambda)b_i(o_T) = \sum_{i=1}^{N}\left(b_i(o_T)\sum_{j=1}^{N} \Pr(o_{1:T-1}, Q_{T-1} = S_j|\lambda)A_{ji}\right)$
- ✓ 红色部分是什么？
  - 一个规模小一点的相同问题（$T-1$）
  - 但是需要对所有j的可能取值计算
  - 可以通过动态规划解决，但是需要解决比原问题更多数目的小规模子问题
  - 但是，复杂的是，目前牵涉两个数值而不是一个：$\Pr(o_{1:T-1}, Q_T = S_i|\lambda)$和$P(o_{1:T}|\lambda)$
  - 计算的方向应该是什么？

动态规划算法（前向forward算法）

- ✓ $P(o_{1:T}|\lambda) = \sum_{i=1}^{N} \Pr(o_{1:T-1}, Q_T = S_i|\lambda)b_i(o_T) = \sum_{i=1}^{N}\left(b_i(o_T)\sum_{j=1}^{N}\Pr(o_{1:T-1}, Q_{T-1} = S_j|\lambda)A_{ji}\right)$
- ✓ 定义
  - $\alpha_t(i) = P(o_{1:t}, Q_t = S_i|\lambda)$ –含义是？
  - Initialization: $\alpha_1(i) = \pi_i b_i(o_1), \quad 1 \le i \le N$
  - Induction: For $1 \le t \le T-1$

$$\alpha_{t+1}(i) = \left[\sum_{j=1}^{N} \alpha_t(j)A_{ji}\right] b_i(o_{t+1}), \qquad 1 \le i \le N$$

  - Termination (output): $\Pr(o_{1:T}|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$

后向算法backward algorithm

- ✓ 定义$\beta_t(i) = \Pr(o_{t+1:T}|Q_t = S_i, \lambda)$
  - 若在时刻t状态为S_i，将来观测到$o_{t+1:T}$的概率
- ✓ 初始化：$\beta_T(i) = 1, \quad 1 \le i \le N$
- ✓ 反向更新：$t = T-1, T-2, \ldots, 2, 1$

$$\beta_t(i) = \sum_{j=1}^{N} A_{ij} b_j(o_{t+1}) \beta_{t+1}(j), \qquad 1 \le i \le N$$

- ✓ 输出：$\beta_1(i) = \Pr(o_{2:T}|Q_1 = S_i, \lambda)$

$$P(o_{1:T}|\lambda) = \sum_{i=1}^{N} \pi_i b_i(o_1) \beta_1(i)$$

Decoding

发现“最好”的隐变量值

- ✓ 标准1：对于每个时刻，发现其后验概率最大的状态
  - 定义$\gamma_t(i) = \Pr(Q_t = S_i | o_{1:T}, \lambda)$，当观测到输出为$o_{1:T}$时，时刻$t$时隐变量为第$i$个状态的后验概率
  - 那么，对于一个输出序列$o_{1:T}$，选择

$$q_t = \underset{1 \le i \le N}{\operatorname{argmax}} \gamma_t(i), \qquad t = 1,2,\ldots,T$$

  - 可能出现什么问题？
  - 不存在这样的路径$q_{1:T}$

怎样计算γ

- ✓ $\alpha_t(i)\beta_t(i) = P(o_{1:T}, Q_t = S_i|\lambda)$
  - 为什么？

- ✓ 贝叶斯定理

$$\gamma_t(i) = \Pr(Q_t = S_i|o_{1:T}, \lambda) = \frac{\Pr(o_{1:T}, Q_t = S_i|\lambda)}{\Pr(o_{1:T}|\lambda)} = \frac{\alpha_t(i)\beta_t(i)}{\Pr(o_{1:T}|\lambda)}$$

  - $\Pr(o_{1:T}|\lambda) = \sum_{i=1}^{N} \alpha_t(i)\beta_t(i)$ for any t!
  - 三种计算方法计算$P(o_{1:T}|\lambda)$了

- ✓ 或者 1) $\gamma_i = \alpha_t(i)\beta_t(i)$ 2) L1 normalize：$\gamma_i \leftarrow \frac{\gamma_i}{\sum_i \gamma_i}$

寻找最大概率的路径

- ✓ 一共有N^T种可能的路径，有些的概率可能为0
  - 比如通过准则1得到的路径
  - 那么，如果寻找所有可能路径里面概率最大的那个呢？

$$q_{1:T} = \underset{Q_{1:T}}{\operatorname{argmax}} \Pr(Q_{1:T}|o_{1:T},\lambda) = \underset{Q_{1:T}}{\operatorname{argmax}} \Pr(Q_{1:T},o_{1:T}|\lambda)$$

- ✓ Naïve的方法复杂性是N^T，有没有更好的方法？
  - Viterbi方法
  - 猜猜这是一种什么类型的方法？
  - Andrew J. Viterbi，USC的工程学院以其命名

Viterbi decoding

- ✓ $q_{1:T} = \underset{Q_{1:T}}{\operatorname{argmax}} \Pr(Q_{1:T}, o_{1:T}|\lambda)$
- ✓ 定义更多的子问题

$$\delta_t(i) = \max_{Q_{1:t-1}} \Pr(Q_{1:t-1}, Q_t = S_i, o_{1:t}|\lambda)$$

  - 含义：当限定两个条件1）前t个时刻的输出为$o_{1:t}$，2）第t个时刻的隐状态为第i个状态的时候，最佳路径所能取得的最大概率
  - 怎么取得q_t?
    - 用另外一个变量$\psi_t(i)$做记录
  - 怎么从t进展到$t+1$?

两个步骤

- ✓ 从t进展到$t+1$
  - $\delta_{t+1}(i) = \max_j\left(\left[\delta_t(j)A_{ji}\right]b_i(o_{t+1})\right)$
  - $\delta_{t+1}(i)$是概率，如果只需要发现概率最大那个状态，$b_i(o_{t+1})$?
- ✓ 所以在时刻$t+1$，需要用另外一个变量$\psi_t(i)$记录最大概率的路径在时刻t是哪一个状态
  - $\psi_{t+1}(i) = \underset{1\le j\le N}{\operatorname{argmax}}\left(\left[\delta_t(j)A_{ji}\right]\right)$

Viterbi算法

- ✓ 初始化：$\delta_1(i) = \pi_i b_i(o_1), \psi_1(i) = 0, 1 \le i \le N$
- ✓ 递归：$2 \le t \le T$，$1 \le i \le N$

$$\delta_t(i) = \max_{1 \le j \le N} \left(\left[\delta_{t-1}(j) A_{ji} \right] b_i(o_t) \right)$$

$$\psi_t(i) = \operatorname{argmax}_{1 \le j \le N} \left(\left[\delta_{t-1}(j) A_{ji} \right] \right)$$

- ✓ 输出：
  - 最大概率：$P^* = \max_{1 \le i \le n} \delta_T(i)$
  - 时刻T的最佳路径变量：$q_T^* = \operatorname{argmax}_{1 \le i \le N}(\delta_T(i))$
  - 时刻$T-1, T-2, \dots, 2, 1$的最佳路径变量：$q_t^* = \psi_{t+1}(q_{t+1}^*)$

分析

- ✓ 问题1的动态规划$\alpha_{t+1}(i) = \sum_{j=1}^{N} \alpha_t(j) A_{ji}$
- ✓ 问题2的动态规划$\delta_t(i) = \max_j \left(\left[\delta_{t-1}(j) A_{ji} \right] b_i(o_t) \right)$
- ✓ 最重要的操作分别是sum-product和max-product
  - 其复杂性均为N^2T
  - 和naïve方法的TN^T比较，极其巨大的速度提高

- ✓ 进一步阅读：sum-product和max-product是更为通用的算法，在图模型graphical model中有极为广泛的应用。

问题3：学习系统的参数

- ✓ 发现$\lambda = (A, B, \boldsymbol{\pi})$，使得对于固定的$N$，$T$，和观察值$\boldsymbol{O}$，似然(likelihood) $P(\boldsymbol{O}|\lambda)$最大

  - 目前没有方法能发现全局最优的解
  - 常用的方法是Baum-Welch算法，发现一个局部最优的解
  - Baum-Welch算法的细节请见教材

  - 进一步阅读，Baum-Welch是EM方法的一种具体体现，更多内容可参考第十四章的习题

Image transformation

What is an image?

What is an image?

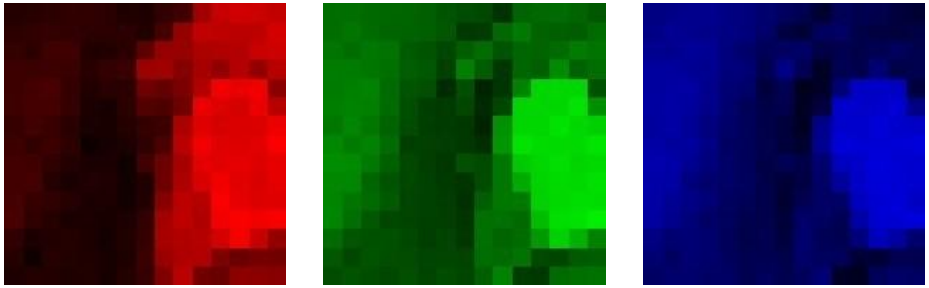
A (color) image is a 3D tensor of numbers.

What is an image?

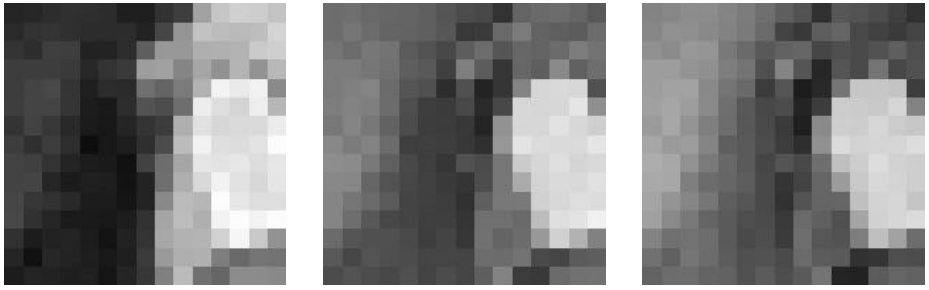
color image patch

How many bits are the intensity values?

red green blue



colorized for visualization



actual intensity values per channel

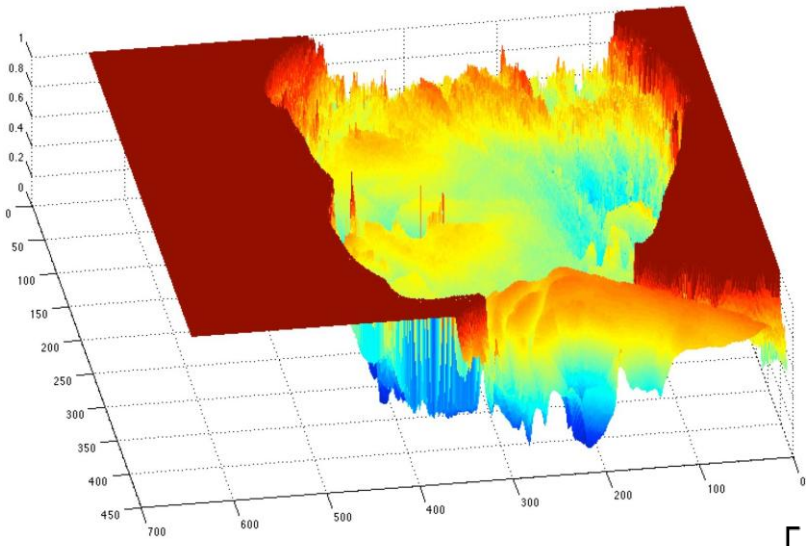
Each channel is a 2D array of numbers.

What is an image?

grayscale image

What is the range of the image function f?

$f(\boldsymbol{x})$



domain $\boldsymbol{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

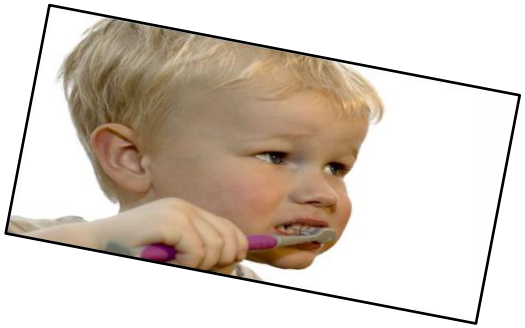
A (grayscale) image is a 2D function.

What types of image transformations can we do?

Filtering

changes pixel *values*

Warping



changes pixel *locations*

What types of image transformations can we do?

F

Filtering

$G(\boldsymbol{x}) = h\{F(\boldsymbol{x})\}$

G

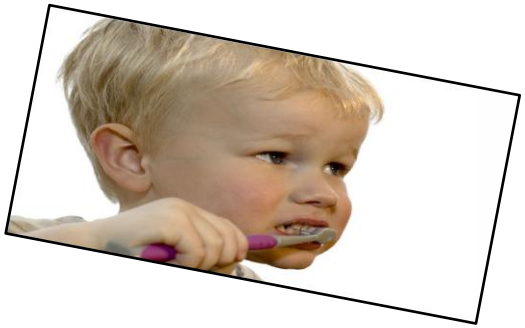
changes *range* of image function

F

Warping

$G(\boldsymbol{x}) = F(h\{\boldsymbol{x}\})$

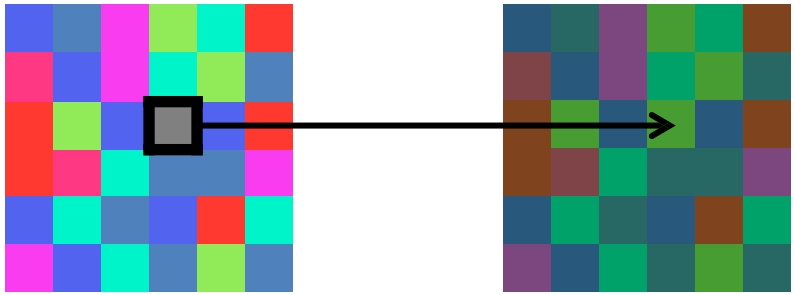
G



changes *domain* of image function

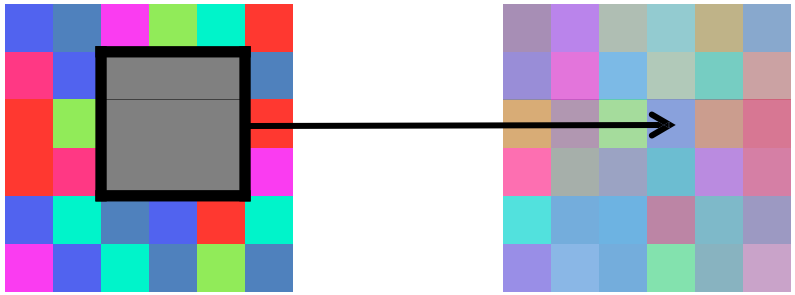
What types of image filtering can we do?

Point Operation



point processing

Neighborhood Operation



“filtering”

Point processing

Examples of point processing

original

darken

lower contrast

non-linear lower contrast

invert

lighten

raise contrast

non-linear raise contrast

How would you implement these?

Examples of point processing

original

x

darken

$x - 128$

lower contrast

$\frac{x}{2}$

non-linear lower contrast

$\left(\frac{x}{255}\right)^{1/3} \times 255$

invert

$255 - x$

lighten

$x + 128$

raise contrast

$x \times 2$

non-linear raise contrast

$\left(\frac{x}{255}\right)^{2} \times 255$

Many other types of point processing

camera output

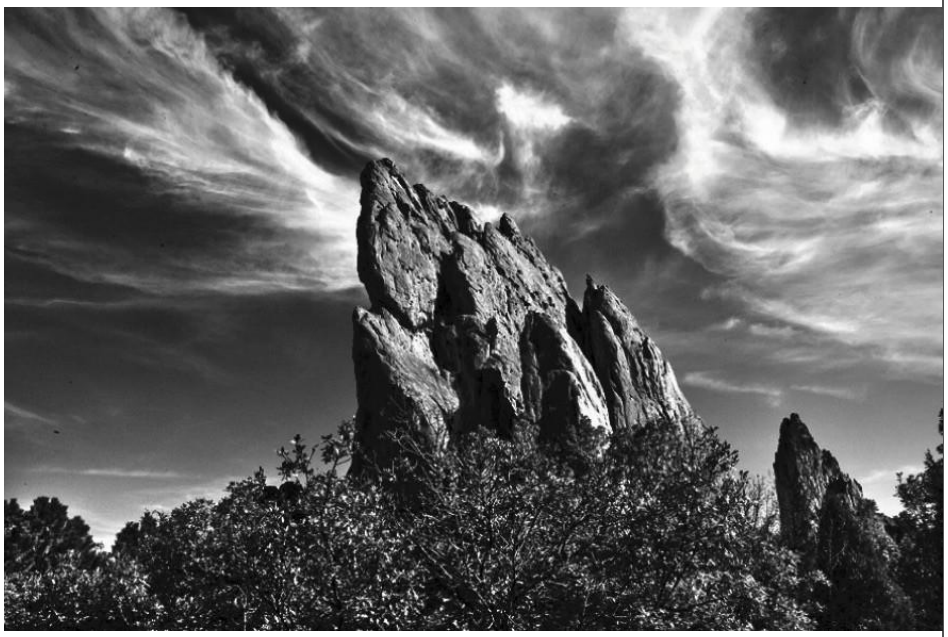


image after stylistic tonemapping

[Bae et al., SIGGRAPH 2006]