## 0.1 Hitchcock Transport Problem

- *Algorithm:* Minimum Cost Flow (algo. 1)
- *Input:* an array of $m$ suppliers, an array of $n$ receivers, a two-dimensional array of cost from $m_i$ to $n_j$.
- *Complexity:* $\mathcal{O}(mn^2(\log m + n\log n))$.
- *Data structure compatibility:* N/A
- *Common applications:* Solve the minimum cost of transportation while meet the requests.

**Problem.** Hitchcock Transport Problem

Suppose there are $m$ sources of commodity $x_1, x_2, \dots, x_m$ with $a(x_i)$ units of supply at $x_i$, and $n$ sinks $y_1, y_2, \dots, y_n$, with demands $b(y_j)$ at $y_j$. Denote $a(x_i, y_j)$ as the unit cost of transportation from $x_i$ to $y_j$, find the solution that meet demands with supplies, while minimizing the transportation cost [3].

### Description

In the "Hitchcock Transport Problem"(abbreviated as HTP), the combination of suppliers and demanders, along with the transportation cost between them, can be seen as a bipartite graph. Therefore, we make some definitions below,

A *flow network* can be represented as a quadratuple $(G, b, u, c)$, where $G$ is a directed graph with $V$, the vertex set, and $E$, the edge set. $b$ is a mapping $V \to \mathbb{R}$ with $\sum_{v\in V} b(v) = 0$, which means that the total flow leaving or entering the network should be none. $u$ is the a *capacity function* with the mapping $E \to \mathbb{R}^+$. $c$ is a *cost function* $E \to \mathbb{R}^+ \cup \{0\}$ [1].

By this construction, our goal is to find the flow $f$ which minimizes $c(f) := \sum_{e\in E} f(e)c(e)$, given the flow network $(G, b, u, c)$.

Also define the set of supplier nodes as $X$, set of demanders as $Y$. Since it would be a necessity to control the nodes $x \in X$ with multiple outgoing edges with positive flow, with setting $f$ as the solution to the HTP, define $\tau_f(x) = |\{y \in Y \mid f(x, y) > 0\}|$ be the number of outgoing edges for some $x$, and $F_f = \{x \in X \mid \tau_f(x) > 1\}$ be the number of supplier nodes $x$ that have multiple outgoing edges.

Vygen [4] has shown that for constant $n$, an optimal solution $f$ can be transformed into an optimal solution $g$ in linear time with $|F_g| \le n - 1$.

Then given a flow network $(G, b, u, c)$, an optimal solution $f$, and $F_f$, we can transform $f$ in $\mathcal{O}(n^2|F_f|)$ time into an optimum solution $g$.

Moreover, according to Ford and Fulkerson [2], if we have an instance $(G, b, u, c)$ with finite capacities on $\sigma$ edges, there exists an equivalent "uncapacitated minimum cost flow instance" [1] with $|V| + \sigma$ nodes and $|E| + \sigma$ edges. Also, denote $M_i(y) = \{x \in X \mid f((x, y)) = b(x)\}$.

Then applying the **Minimum Cost Flow** 1, we can get the minimum cost in $\mathcal{O}(mn^2(\log m + n\log n))$ time.

**Algorithm 1:** Minimum Cost Flow

**Input :** A flow network instance $(G, b, u, c)$
**Output:** The minimum cost flow $f$

**foreach** *edge e in E* **do**
  $f(e) \leftarrow 0$;
**end foreach**
Sort the nodes in $X$ such that $X = x_1, x_2, \ldots, x_m$ with $b(x_1) \geq b(x_2) \geq \cdots \geq b(x_n)$;
**for** $i = 1$ *to* $n$ **do**
  Construct a new flow network $NW_i = (G^i, b^i, u^i, c^i)$ as stated above;
  Compute a minimum cost flow $g$ in $NW_i$ such that the edges with positive flow do not form a cycle;
  **for** $e = (v, w) \in E(G^i)$ *and* $g(e) > 0$ **do**
    /* $e \in X \times Y$ means that $v \in X$ and $w \in Y$ */
    **if** $e \in X \times Y$ **then**
      $f(e) \leftarrow f(e) + g(e)$;
    **end if**
    **if** $e \in Y \times F_f$ **then**
      /* update the backward edges */
      $f((w, v)) \leftarrow f((w, v)) - g(e)$;
    **end if**
    **if** $e \in Y \times Y$ **then**
      Select $x \in M_i(x)$ with $c^i(e) = c((x, w)) - cost((x, v))$;
      $f((x, v)) \leftarrow f((x, v)) - g(e)$;
      $f((x, w)) \leftarrow f((X, y)) + g(e)$;
    **end if**
  **end for**
  Retransform $f$ according to Ford and Fulkerson;
**end for**
**return** $f$;

## References.


[1] Ulrich Brenner. "A faster polynomial algorithm for the unbalanced Hitchcock Transportation Problem". In: *Operations Research Letters* 36.4 (2008), pp. 408–413. DOI: 10.1016/j.orl.2008.01.011 (cit. on p. 1).

[2] Lester Randolph Ford and Delbert Ray Fulkerson. *Flows in networks*. Princeton University Press, 1962 (cit. on p. 1).

[3] Frank L. Hitchcock. "The distribution of a product from several sources to numerous localities". In: *Journal of Mathematics and Physics* 20.1-4 (1941), pp. 224–230. DOI: 10.1002/sapm1941201224 (cit. on p. 1).

[4] Jens Vygen. "Geometric quadrisection in linear time, with application to VLSI Placement". In: *Discrete Optimization* 2.4 (2005), pp. 362–390. DOI: 10.1016/j.disopt.2005.08.007 (cit. on p. 1).


## 0.2 Image Enhancement

- *Algorithm:* Contrast Enhancement (algo. 2)
- *Input:* An image with $m \times n$ pixels.
- *Complexity:* $\mathcal{O}(mn)$
- *Data structure compatibility:* N/A
- *Common applications:* Make images which are overexposed or underexposed better to recognize, or enhance images with too dark or too bright background.

**Problem.** Image Enhancement

Sometimes an image would be too hard for people to recognize some important details in it due to overexposure, underexposure, or background light problems. Therefore, how to enhance the image, such that the details can be highlighted and clear to see would be the problem.

## Description

There are many ways to realize image enhancement. This time I would focus on the contrast enhancement method which takes use of grey-level histogram.

In image editing and manipulation, contrast enhancement is the method to adjust the histogram of an image to shift pixels between the lightest and darkest parts of the image [1].

Usually, contrast enhancement is performed in two steps, firstly a tonal enhancement and then the contrast stretch. Tonal enhancements "improve the brightness differences in the dark", grey and bright regions, while a contrast stretch increases the brightness differences "uniformly across the dynamic range of the image. [2]"

### Grey-level($GL$) Histogram

Once given an image, a corresponding grey-level histogram will be created by counting how many times each grey-level value (from 0 to 255) are present in the image. For example, Fig. 1 shows the grey-level histogram of an image.

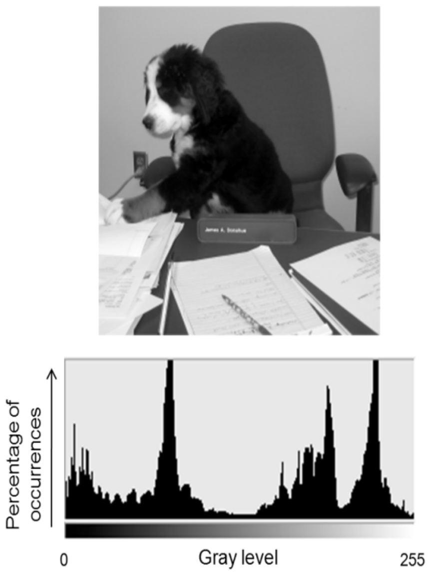


Figure 1: The grey-level histogram of an image

After getting the grey-level histogram, a mapping function would be applied to adjust the current grey-level $GL$ to a new one $GL'$ to realize enhancement. The simplest method is directly contrast reversal, which means that for every pixel in the image, the grey-level value $g(x, y)$ ($(x, y)$ denotes the location of the pixel) would be mapped into $g'(x, y) = 255 - g(x, y)$, the effect of which can be illustrated by Fig. 2.

### Contrast Stretch

A high-contrast easy-to-recognize image contains gray-level values of the full range 0 to 255. One method is to create a remapping such that the lowest $GL_{min}$ will be mapped to 0 and the highest $GL_{max}$ will be remapped to 255. Also, $GL$ between them would be linearly remapped between 0 and 255, with the transformation equation as

$$g'(x, y) = \lfloor \frac{255}{GL_{max} - GL_{min}} (g(x, y) - GL_{min}) \rfloor$$

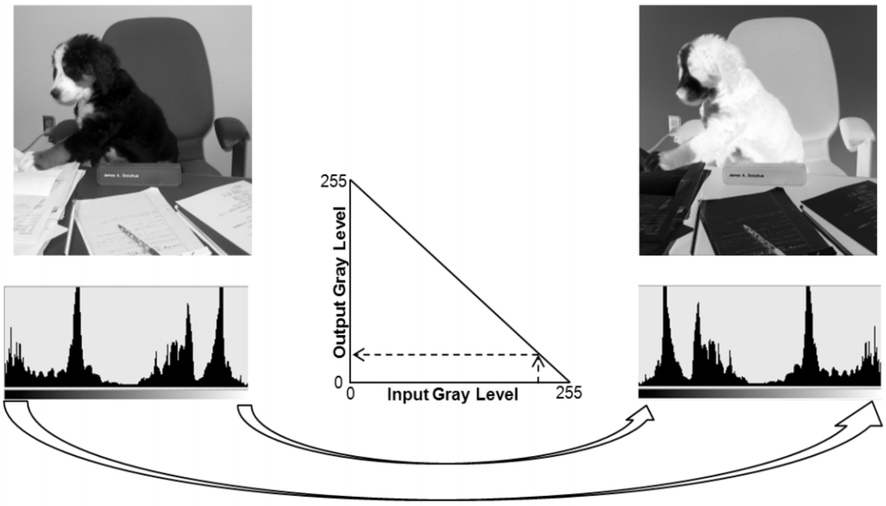


Figure 2: Contrast Reversal

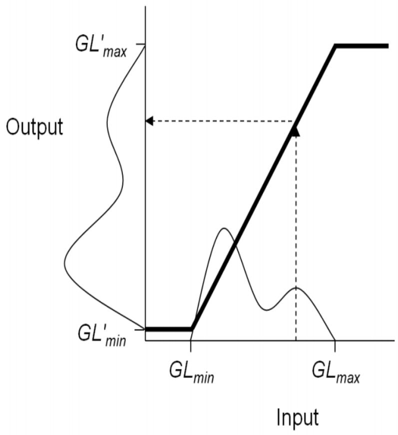


Figure 3: Linear Transformation of the grey-level value

If we want to customize the remapped $GL'_{min}$ and $GL'_{max}$, then we can create the following generalized linear transformation.

$$g'(x,y) = \lfloor \frac{GL'_{max} - GL'_{min}}{GL_{max} - GL_{min}}(g(x,y) - GL_{min}) + GL'_{min} \rfloor$$

In this way, the relative shape of the histogram would be unchanged, but the range would be expanded to fill all between $[GL'_{min}, GL'_{max}]$. The transformation function is shown in Fig. 3. And the comparison between the original image and the image after stretch is depicted in Fig. 4.

### Tonal Enhancement

The contrast stretch method only deal with linear transformation of grey-level values of an image. Sometimes we may need nonlinear transformations such that we can enhance some grey-level regions while potentially reduce the contrast in other regions [2].

The most common nonlinear transformation is the gamma correction [3], which uses an exponent of $1/\gamma$ to preprocess the image to produce a linear response to brightness. The transformation formula is as follows

$$g'(x,y) = \lfloor 255 \cdot (\frac{g(x,y)}{255})^{\frac{1}{\gamma}}) \rfloor$$

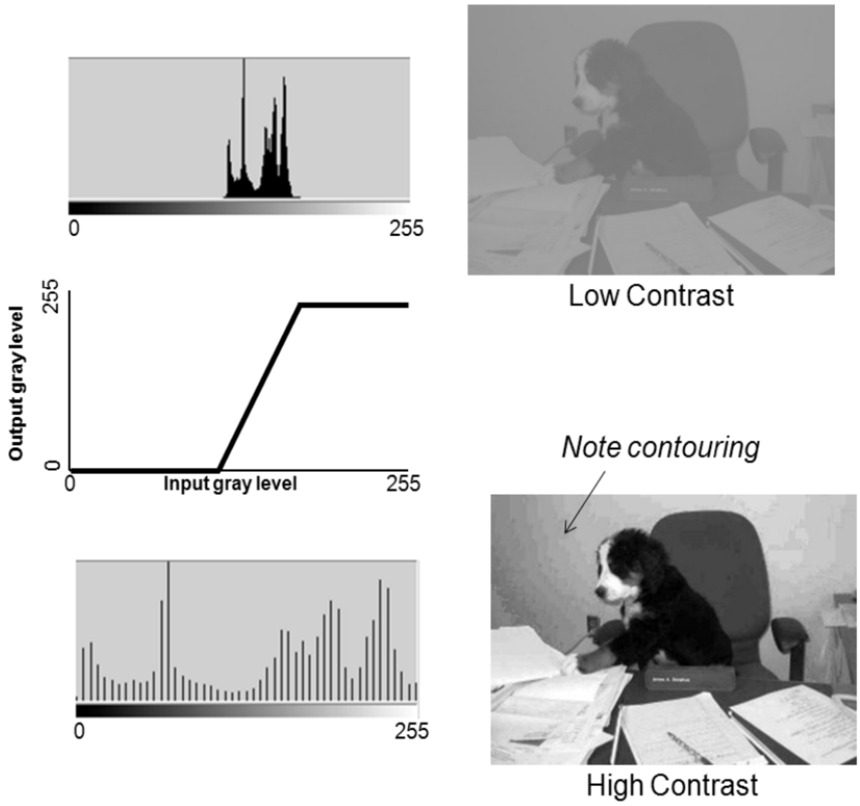


Figure 4: Enhancement by contrast stretch

If we want to enhance very dark images, the logarithmic enhancement method can be applied as

$$g'(x,y) = \lfloor 255 \cdot \frac{\log(g(x,y)+1)}{255} \rfloor$$

Fig. 5 and Fig. 6 would show the effect of gamma correction and logarithmic enhancement.

---

**Algorithm 2:** Contrast Enhancement

**Input :** an Image $I$ with $m \times n$ pixels
**Output:** the corresponding image after enhancement $I'$

hist $\leftarrow$ an array of 0s of size 256;
**for** $i = 1$ *to* $m$ **do**
  **for** $j = 1$ *to* $n$ **do**
    $GL \leftarrow$ the grey-level value $g(i,j)$;
    hist[$GL$] $++$;
  **end for**
**end for**
Choose a specific method of enhancement $M$;
hist_after $\leftarrow$ an array of 0s of size 256;
**for** $i = 1$ *to* $m$ **do**
  **for** $j = 1$ *to* $n$ **do**
    $GL' \leftarrow M(g(i,j))$;
    $I'(i,j) \leftarrow GL'$;
    hist_after[$GL'$] $++$;
  **end for**
**end for**
**return** $I'$;

---

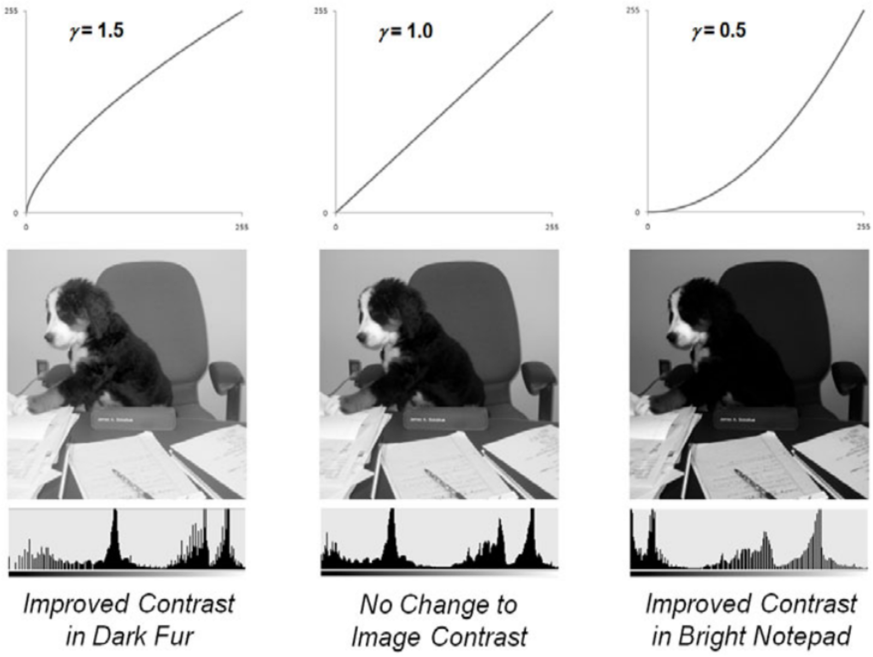


Figure 5: Enhancement using gamma correction

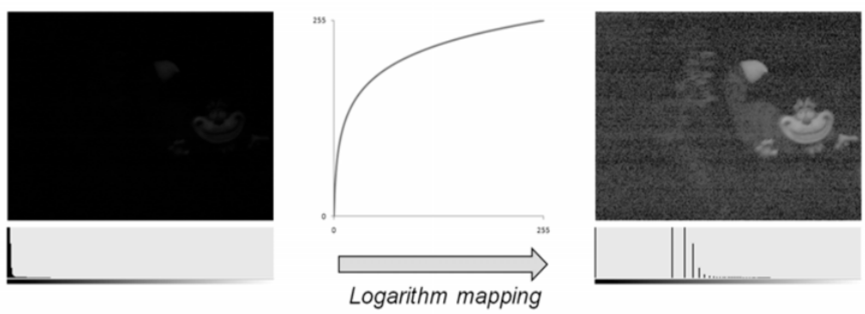


Figure 6: Logarithmic enhancement

## References.


[1] *Contrast enhancement*. URL: http://printwiki.org/Contrast_Enhancement (cit. on p. 3).

[2] Robert D. Fiete. “Image Enhancement Processing”. In: *Modeling the imaging chain of digital cameras*. SPIE Press, 2010, pp. 127–161 (cit. on pp. 3, 4).

[3] Charles A. Poynton. Morgan Kaufmann Publishers, 2003, p. 260 (cit. on p. 4).