# Functors, Comonads, and Digital Image Processing

Le, Justin A. (Advisor: Dr. Mohamed Allali)
Chapman University, Schmid College of Science and Technology

## Objectives

Much work has been done recently on designing abstract data structures and library interfaces modeled on principles of category theory. While much benefit has been shown in program architecture, there is promise in applying these principles to the fields of digital image processing and numerical algorithms in general. We will:

- Build an abstraction on digital image processing filters and operations involving structures from abstract algebra and category theory.
- Explore its usage as an actual, implemented API
- Consider generalizations of these basic principles to numerical algorithms as a whole.

## Endofunctors and Comonads

The main category we are considering is **Set** ($\mathcal{S}$), where the objects are sets and the morphisms between sets $X$ and $Y$ are the typical functions $f : X \to Y$.
An **endofunctor** $F$ on $\mathcal{S}$ is a *category homomorphism* on $\mathcal{S}$, mapping a set $X$ to a set $FX$ and all functions $f : X \to Y$ to $Ff : FX \to FY$, preserving categorical structure. A **natural transformation** is a *functor homomorphism*.
A **comonad** is a *comonoid* in the category of endofunctors.
A comonad $W$ on category $\mathcal{C}$ is a triple $\langle W, \epsilon, \delta \rangle$, with:

- An endofunctor $W : \mathcal{C} \to \mathcal{C}$
- A natural transformation $\epsilon : W \to 1_{\mathcal{C}}$, called "counit" or "extract", where $1_{\mathcal{C}}$ is the identify functor on $\mathcal{C}$.
- A natural transformation $\delta : W \to W^2$, called "comultiply" or "duplicate".

Such that $\epsilon$ and $\delta$ form a comonoid as counit and comultiply.
For a particular comonad, it is often useful to consider its **cokleisli category**. The cokleisli category $\mathcal{C}_W$ formed by a comonad $W$ on category $\mathcal{C}$ is a category with the same objects as $C$, but morphisms $f : WX \to Y$, with source $WX$ in $\mathcal{C}$, are considered morphism with source $X$ in $\mathcal{C}_W$.
Surprisingly, these form a valid category iff $W$ is a comonad:

$$g \circ_W f \equiv g \circ Wf \circ \delta \tag{1a}$$
$$1_X = \epsilon_X \tag{1b}$$

That $\mathcal{C}_W$ forms a category is of great significance and is the entire foundation of this presentation's findings. For any comonad $W$, we compose $f : WX \to Y$ and $g : WY \to Z$ as $f \circ_W g : WX \to Z$. We can use this result to also *construct* a comonad through valid categories of this shape, with $\delta_{W(X)} = 1^{\mathcal{C}}_X \circ_W 1^{\mathcal{C}}_X$.
We also define **cokleisli extension**, lifting $f : WX \to Y$ to $f^* : WX \to WY$:

$$f^* \equiv Wf \circ \delta \tag{2}$$

## "Image with Focus" Functor

Consider the endofunctor $I_n$ on $\mathcal{S}$:

$$I_n(X) = X^{\mathbb{Z}^n} \times \mathbb{Z}^n \tag{3a}$$
$$I_n(f) = \lambda(h, \mathbf{c}).\ (f \circ h, \mathbf{c}) \tag{3b}$$

$$x \qquad \left( \begin{bmatrix} \ddots & \vdots & \vdots & \vdots & \unicode{x22F0} \\ \cdots & x_{0,0} & x_{0,1} & x_{0,2} & \cdots \\ \cdots & x_{1,0} & x_{1,1} & x_{1,2} & \cdots \\ \cdots & x_{2,0} & x_{2,1} & x_{2,2} & \cdots \\ \unicode{x22F0} & \vdots & \vdots & \vdots & \ddots \end{bmatrix}, (2,3) \right)$$

Figure 1: A member of set $X$ (left) and set $I_2(X)$ (right)

$I_2$ takes a set $X$ to an *array* of elements of $X$, with a "focus" index. On $[0, N] \subset \mathbb{N}$, one gets the set of *grayscale* images whose pixels take on intensities in $[0, N]$, with a focus.

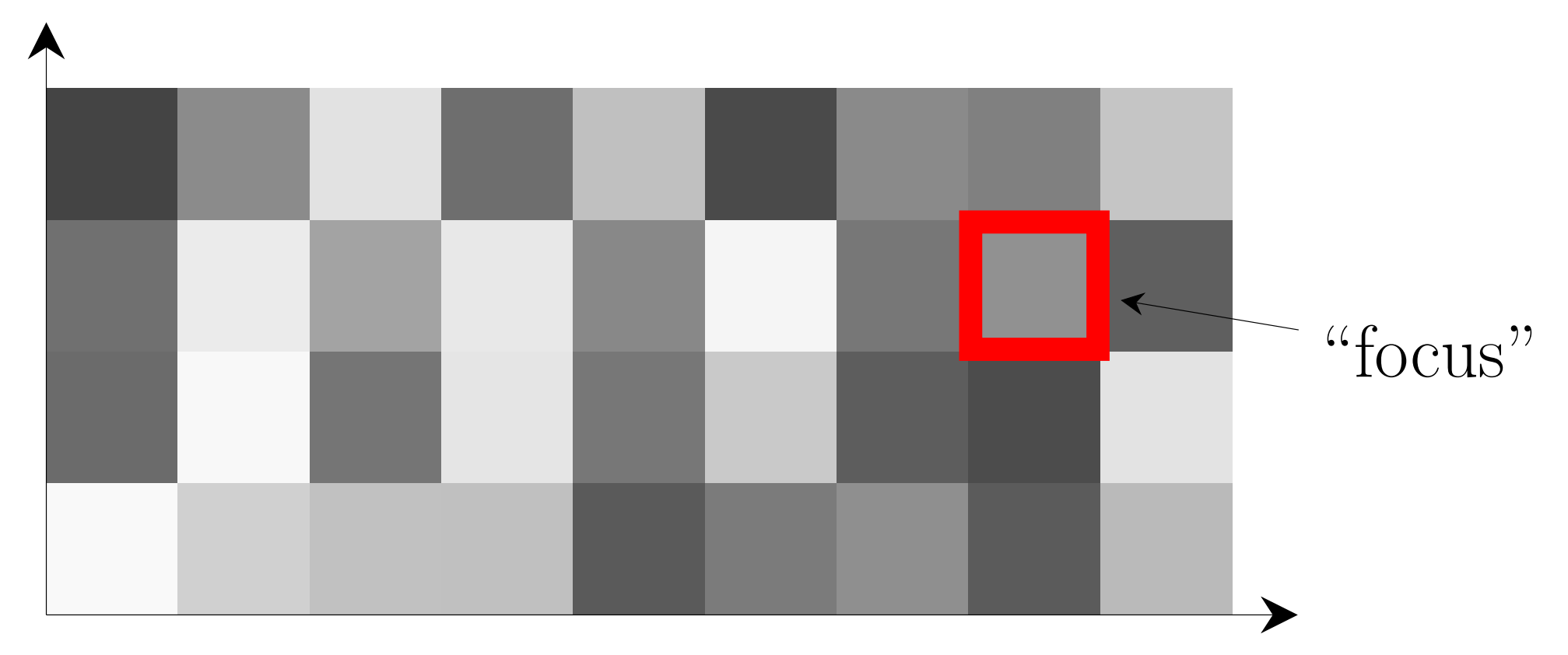

Figure 2: Visualization of a member of $I_2([0, N] \subset \mathbb{N})$

## Cokleisli Affine Transformations

We see *cokleisli arrows* on $I_n$ as functions that, given a "whole image" returns a "new value at the focus". A rotation would take $(h, \mathbf{c})$ and give the value at coordinate $\mathbf{c}$ after rotating $h$. (See fig. 6(a))
Affine transformations describe scalings, translations, skewings and rotations and in $\mathbb{R}^2$ can be encoded with augmented matrices.

$$\left[\begin{array}{cc|c} 0 & -1 & w \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{array}\right]$$

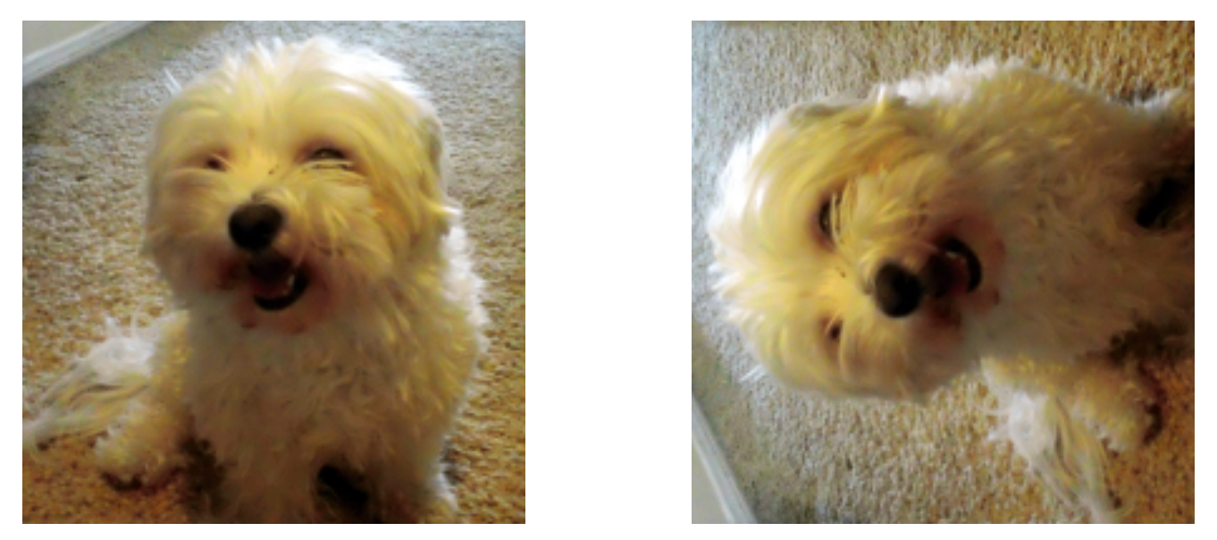

Figure 3: An affine transformation matrix and its encoded effect

Affine transformations with inverses that can be encoded as a $\mathbb{Z}$-valued augmented matrix encode the cokleisli arrow $\alpha_A : I_w(\mathbb{R}) \to \mathbb{R}$, which gives $x$ by applying $A$ to the matrix $h(i, j)$ and indexing into the result with focus $\mathbf{c}$. We recover the original transformation with cokleisli extension (fig. 6).

$$\alpha_A(h, \mathbf{c}) = h(A^{-1}(\mathbf{c})) \tag{4a}$$
$$(g \circ_{I_n} f)(h, \mathbf{c}) \equiv g(\lambda \mathbf{r}.\ f(h, \mathbf{r}), \mathbf{c}) \tag{4b}$$
$$1_X(h, \mathbf{c}) = h(\mathbf{c}) = \alpha_I \tag{4c}$$
$$\alpha_B \circ_{I_n} \alpha_A = \alpha_{B \circ A} \tag{4d}$$

From this we can construct a comonad $\forall n.\ I_n$.

## Local Neighborhood Functor

Endofunctor $G_n$ can represent "local neighborhoods":

$$G_n(X) = X^{\mathbb{Z}^n} \tag{5a}$$
$$G_n(f) = \lambda h.\ f \circ h \tag{5b}$$

$$x \qquad \begin{bmatrix} \ddots & \vdots & \vdots & \vdots & \unicode{x22F0} \\ \cdots & x_{-1,-1} & x_{-1,0} & x_{-1,1} & \cdots \\ \cdots & x_{0,-1} & x_{0,0} & x_{0,1} & \cdots \\ \cdots & x_{1,-1} & x_{1,0} & x_{1,1} & \cdots \\ \unicode{x22F0} & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Figure 4: A member of set $X$ (left) and set $G_2(X)$ (right)

## Kernel Filters as Cokleisli Arrows

"**Kernels**" (or *convolution matrices*, or *masks*) are an encoding of local *linear* filters used for blurring, sharpening, and edge-detection.

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

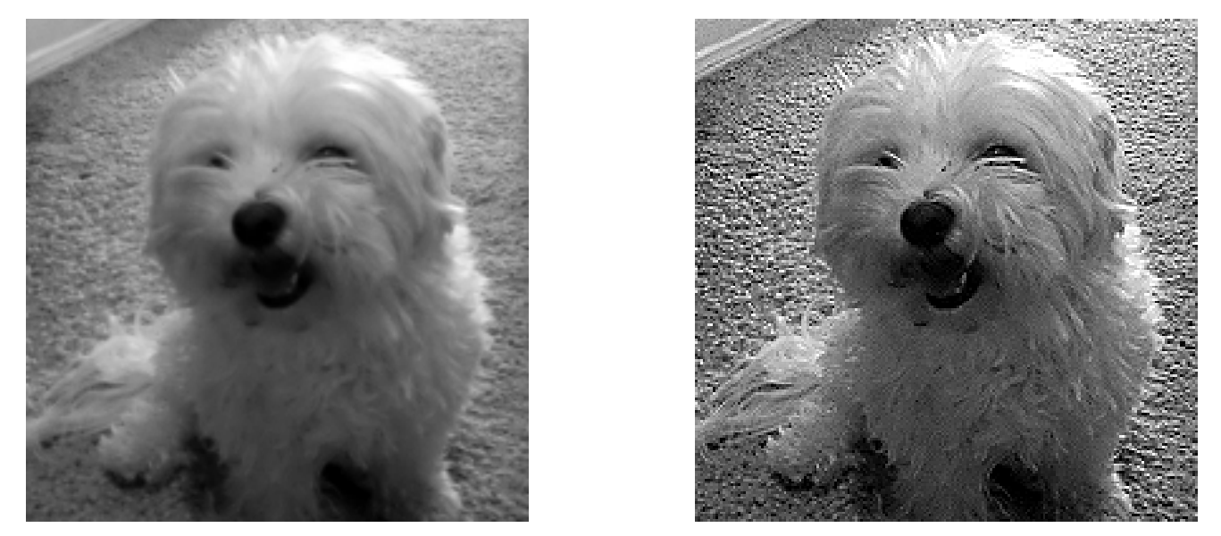

Figure 5: A "kernel" and the effect of the filter it encodes

If we take the kernel center to be at index $(0, 0)$, a kernel $k$ actually encodes a *cokleisli arrow*, $\kappa_k : G_2(\mathbb{R}) \to \mathbb{R}$:

$$\kappa_k(h) = \sum_{i \in \mathbb{Z}} \sum_{j \in \mathbb{Z}} k_{ij} h(i, j) \tag{6}$$

That is, it maps every **neighborhood** to a **single value**, by summing all of the neighborhood values together weighted by the $k$. Such $\kappa_k$, $\kappa_j$ compose under convolution $*$, with identity morphism $1_X$: (and thus yields a comonad)

$$(g \circ_{G_n} f)(h) \equiv g(\lambda \mathbf{r}.\ f(\lambda \mathbf{s}.\ h(\mathbf{r} + \mathbf{s}))) \tag{7a}$$
$$1_X(h) = h(\mathbf{0}) = \kappa_\delta \quad \text{(dirac delta)} \tag{7b}$$
$$\kappa_j \circ_{G_n} \kappa_k = \kappa_{j*k} \tag{7c}$$

## Nonlinear generalizations

Affine transformations and kernel filters can only encode linear filters, but the full expression and encoding of cokleisli arrows in $I_n$ and $G_n$ allows for *non-linear* filters as well, such as kinetic energy operator $f : G_1(\mathbb{R}) \to \mathbb{R}$:

$$f(h) = \frac{1}{2}\, m \left( \frac{h(1) - h(0)}{\Delta t} \right)^2 \tag{8}$$

## The Power of Extension

It can be shown that extension of cokleisli arrows of $I_n$ always leave $\mathbf{c}$ unchanged. For arbitrary $\mathbf{c} \in \mathbb{Z}^n$, this means we can consider extensions as morphisms $X^{\mathbb{Z}^n} \to Y^{\mathbb{Z}^n}$ in *absolute position space*, which is something we can use as an **actual image filter**.

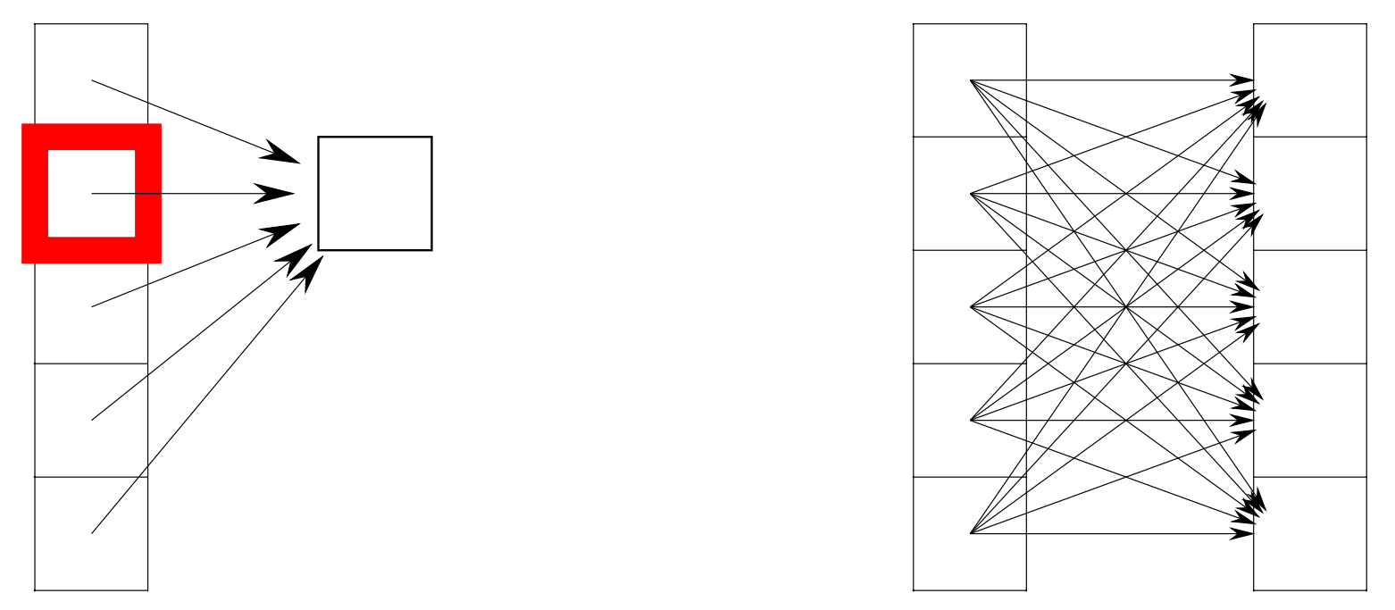

Figure 6: Action of extension on cokleisli arrows of $I_1$

$G_n$ is not as straightforward because there isn't enough information to encode arbitrary transformations. However, choosing a globalization scheme, we can create a functor $L : \mathcal{S}_{G_n} \to \mathcal{S}_{I_n}$, from which extension yields our goal.
In general, filters encoded in $\mathrm{Hom}_{\mathcal{S}_{I_2}}(X, Y)$ and $\mathrm{Hom}_{\mathcal{S}_{G_2}}(X, Y)$ are restricted compared to arbitrary $X^{\mathbb{Z}^n} \to Y^{\mathbb{Z}^n}$. Taking advantage of filters encoded as cokleisli arrows, we open many previously unaccessible avenues through encoding filters as arbitrary $X^{\mathbb{Z}^n} \to Y^{\mathbb{Z}^n}$; they are:

- **Computationally efficient** and **generically optimizable**
- Amenable to **static analysis** and **compiler optimization**
- **Implicitly parallelizable** with **high efficiency**

## Conclusion

The power of this approach lies in ten main advantages:

- **Restricted encodings** of filters as cokleisli arrows
- A **semantic framework** for manipulating and composing such filters
- An efficient **functor action** on cokleisli arrows
- The ability to efficiently **compose** cokleisli arrows
- The ability to efficiently **extend** cokleisli arrows

An implementation of both linear and nonlinear filters as cokleisli arrows leads opens doors to static analysis yielding *compiler optimization* and *implicit parallelism* that are inaccessible when encoding filters in the traditional way.
These benefits are not merely theoretical, but yield real computational benefits.
Furthermore, our framework and potential implementation is generic enough to be directly applicable to $I_n$ and $G_n$ for all $n \in \mathbb{N}$. These techniques can be lifted directly to contexts of digital signal and audio processing with $n = 1$, discretized PDE solvers in $n \in [1, 4]$, and any numerical domain where *local neighborhoods* and *global contexts* and their cokleisli arrows are important objects to analyze.