FFTs in Graphics and Vision

Alignment, Invariance
and Pattern Matching

Outline

Alignment

Shape Matching

Invariance

Pattern Matching

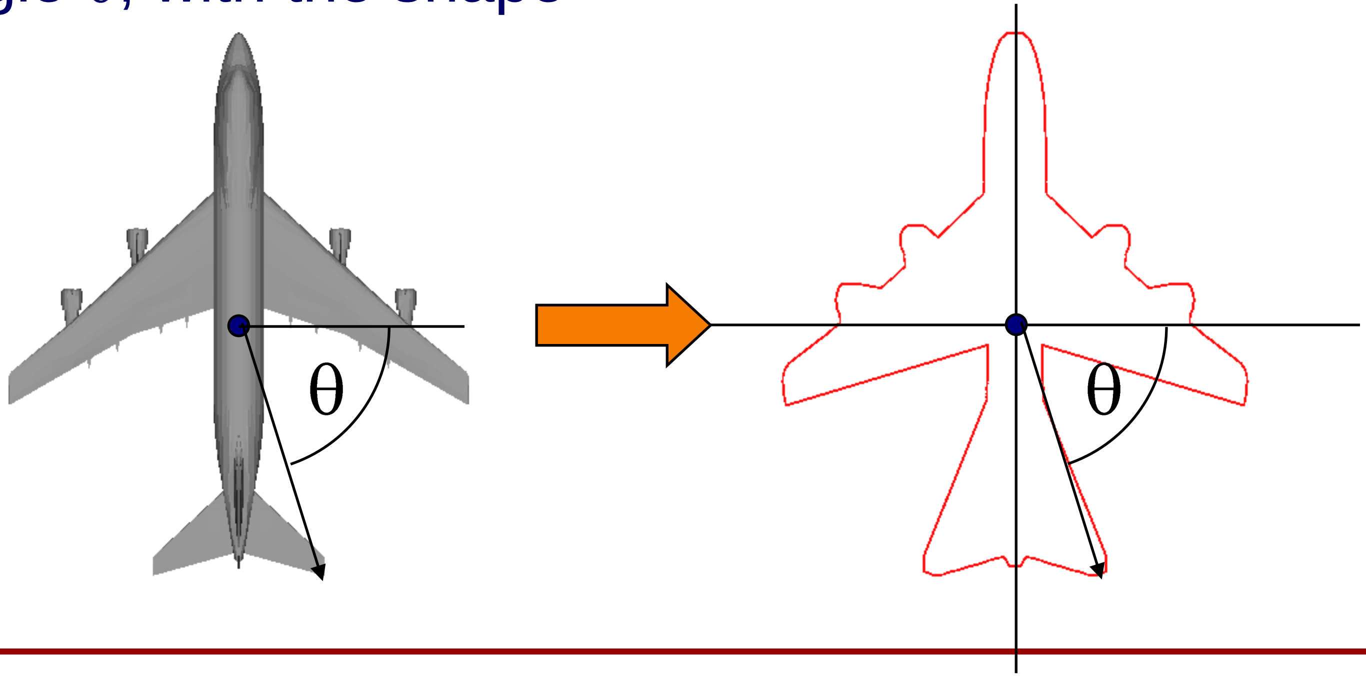
Shape Representation

For 2D shape matching/analysis, it is common to represent the geometry of a shape by a circular array of real values.

Shape Representation

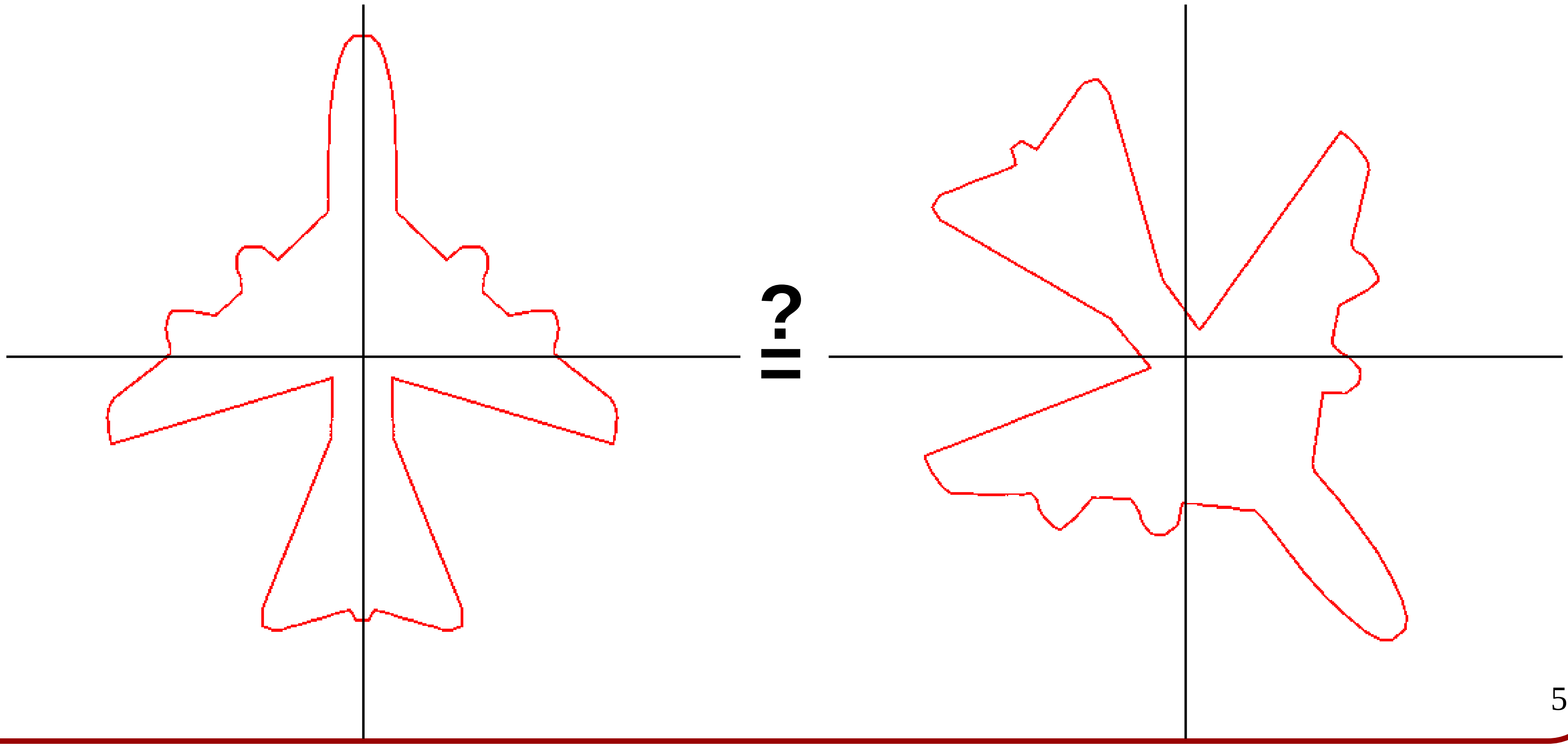
Example:
The circular extent function represents the extent of the shape about the center of mass:

- The value at an angle θ is the distance to the last point of intersection of the ray from the origin, with angle θ, with the shape



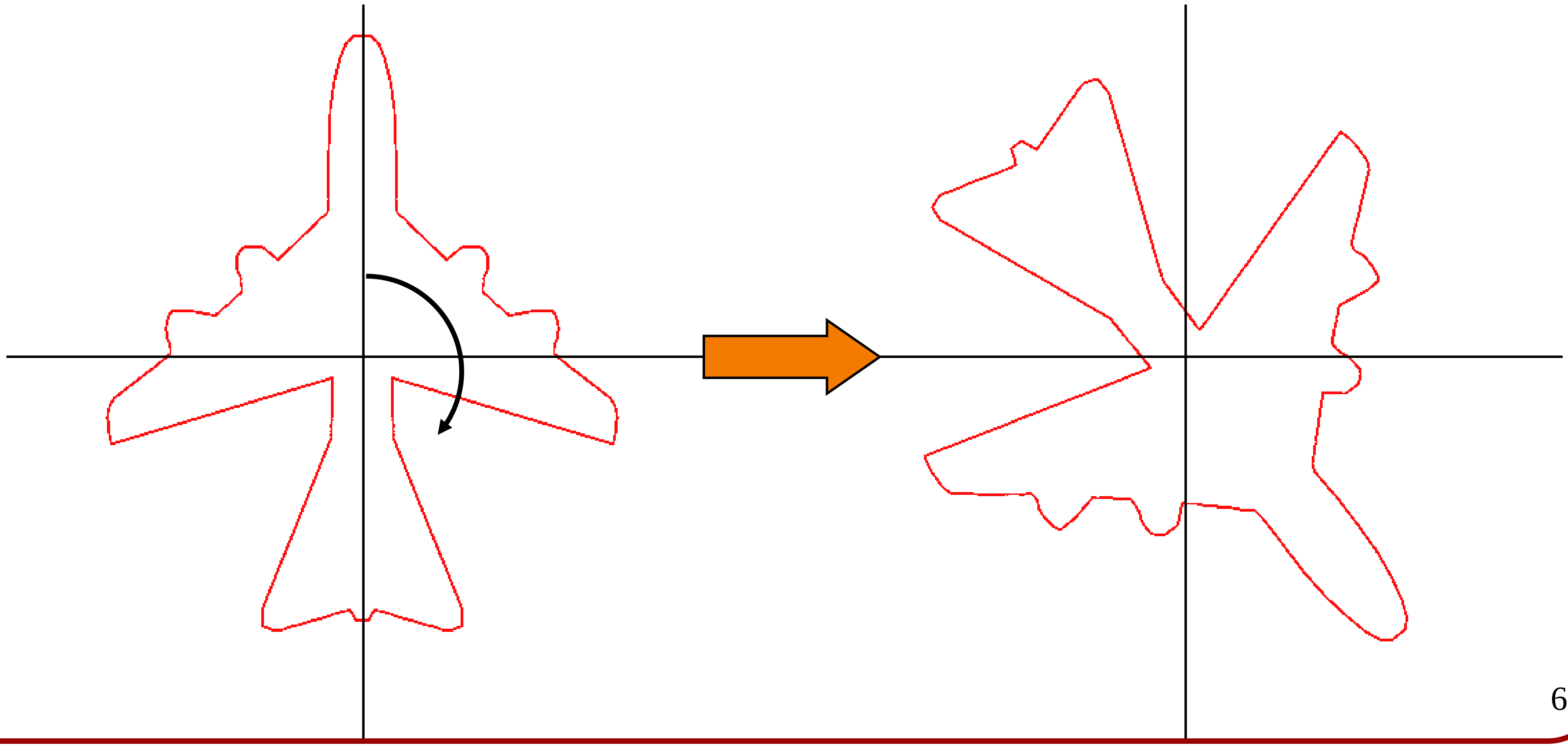
Alignment

Since the shape of an object doesn't change when we rotate it, we would like to know if the two arrays are equivalent, up to rotation.



Alignment

Is there a rotation that will rotate the first array into the second?



Alignment

Is there a rotation that will rotate the first array into the second?

Given the *n*-dimensional arrays *f*[ ] and *g*[ ], is there an index α such that:

$$g[\,] = \rho_\alpha(f[\,])$$

$$\Updownarrow$$

$$g[k] = f[k-\alpha] \qquad \forall\, 0 \le k < n$$

Semantics

In a continuous setting, asking the binary question “are the arrays equal” is not very meaningful, since

- Sampling
- Noise
- Etc

can result in two “equal” arrays having different values.

Semantics

Is there a rotation that will rotate the first array into the second?

$\Downarrow$

For every rotation, how close is the rotation of the first array to the second array?

Semantics

Is there a rotation that will rotate the first array into the second?

For every rotation, how close is the rotation of the first array to the second array?

For every rotation α, what is the value of:

$$D\left(\rho_\alpha(f[\,]), g[\,]\right)$$

Alignment

Since the space of functions on a circle is an inner product space, there is an implicit metric defined by:

$$D^2\left(f[\ ],g[\ ]\right)=\left\|f[\ ]-g[\ ]\right\|^2=\left\langle f[\ ]-g[\ ],f[\ ]-g[\ ]\right\rangle$$

Alignment

Since the space of functions on a circle is an inner product space, there is an implicit metric defined by:

$$D^2\left(f[\ ],g[\ ]\right)=\left\| f[\ ]-g[\ ]\right\|^2=\left\langle f[\ ]-g[\ ],f[\ ]-g[\ ]\right\rangle$$

So in our situation, we would like to evaluate:

$$D^2\left(\rho_\alpha(f[\ ]),g[\ ]\right)=\left\langle \rho_\alpha(f[\ ])-g[\ ],\rho_\alpha(f[\ ])-g[\ ]\right\rangle$$

at every α.

Alignment

$$\boxed{D^2\left(\rho_\alpha(f[\,]), g[\,]\right) = \left\langle \rho_\alpha(f[\,]) - g[\,], \rho_\alpha(f[\,]) - g[\,] \right\rangle}$$

Re-writing this equation gives:

$$\begin{aligned} D^2\left(\rho_\alpha(f[\,]), g[\,]\right) = {} & \left\langle \rho_\alpha(f[\,]), \rho_\alpha(f[\,]) \right\rangle + \left\langle g[\,], g[\,] \right\rangle \\ & - \left\langle \rho_\alpha(f[\,]), g[\,] \right\rangle - \overline{\left\langle \rho_\alpha(f[\,]), g[\,] \right\rangle} \end{aligned}$$

Alignment

$$\boxed{\begin{aligned} D^2\left(\rho_\alpha(f[\,]),g[\,]\right) = {} & \langle \rho_\alpha(f[\,]),\rho_\alpha(f[\,])\rangle + \langle g[\,],g[\,]\rangle \\ & - \langle \rho_\alpha(f[\,]),g[\,]\rangle - \overline{\langle \rho_\alpha(f[\,]),g[\,]\rangle} \end{aligned}}$$

Since the Hermitian dot-product of two real-valued arrays is also a real value:

$$D^2\left(\rho_\alpha(f[\,]),g[\,]\right) = \|\rho_\alpha(f[\,])\|^2 + \|g[\,]\|^2 - 2\langle \rho_\alpha(f[\,]),g[\,]\rangle$$

Alignment

$$\boxed{D^2\left(\rho_\alpha(f[\,]),g[\,]\right)=\left\|\rho_\alpha(f[\,])\right\|^2+\left\|g[\,]\right\|^2-2\left\langle\rho_\alpha(f[\,]),g[\,]\right\rangle}$$

Since ρ_α is a unitary transformation:

$$D^2\left(\rho_\alpha(f[\,]),g[\,]\right)=\left\|f[\,]\right\|^2+\left\|g[\,]\right\|^2-2\left\langle\rho_\alpha(f[\,]),g[\,]\right\rangle$$

Alignment

$$\boxed{D^2\left(\rho_\alpha(f[\,]), g[\,]\right) = \|f[\,]\|^2 + \|g[\,]\|^2 - 2\langle \rho_\alpha(f[\,]), g[\,]\rangle}$$

So, to compute the distance between a rotation of the circular array *f*[ ] by α, and the circular array *g*[ ], we need to know:

- The magnitude of *f*[ ]: $\|f[\,]\|^2$,
- The magnitude of *g*[ ]: $\|g[\,]\|^2$,
- The value of the cross-correlation: $\langle \rho_\alpha(f[\,]), g[\,]\rangle$

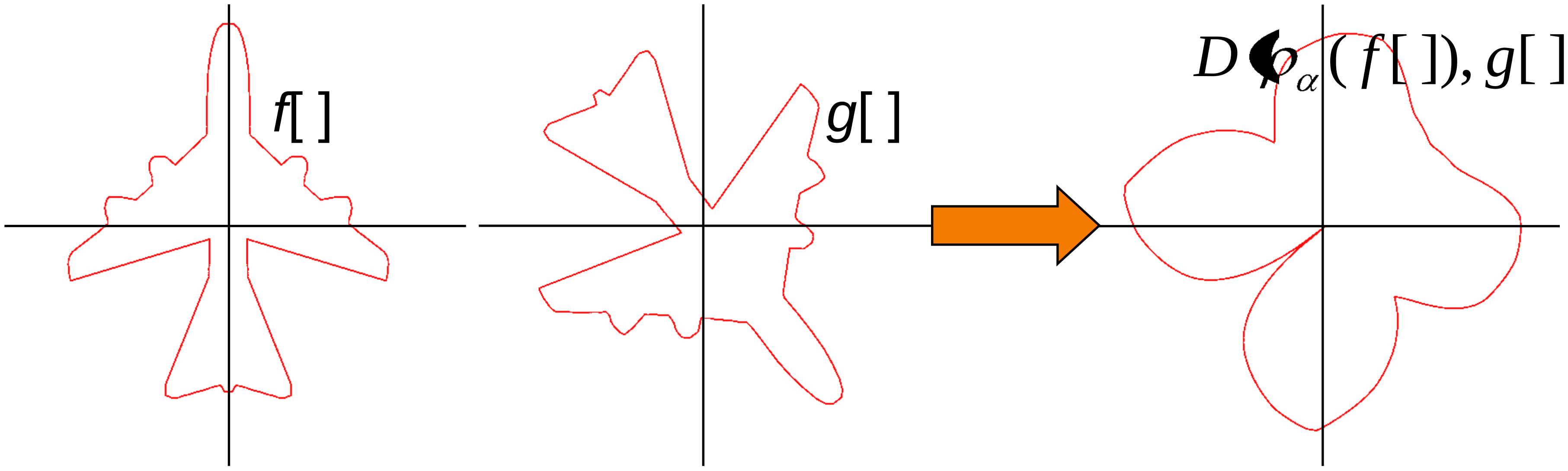
Alignment

$$\boxed{D^2\left(\rho_\alpha(f[\,]),g[\,]\right)=\|f[\,]\|^2+\|g[\,]\|^2-2\langle\rho_\alpha(f[\,]),g[\,]\rangle}$$

- The size of *f*[ ]: $\|f[\,]\|^2$
  - This is a constant value independent of α and can be computed in O(*n*) time.
- The size of *g*[ ]: $\|g[\,]\|^2$
  - This is a constant value independent of α and can be computed in O(*n*) time.
- The value of the cross-correlation: $\langle\rho_\alpha(f[\,]),g[\,]\rangle$
  - This can be computed using the FFT in O(*n* log *n*) time.

Alignment

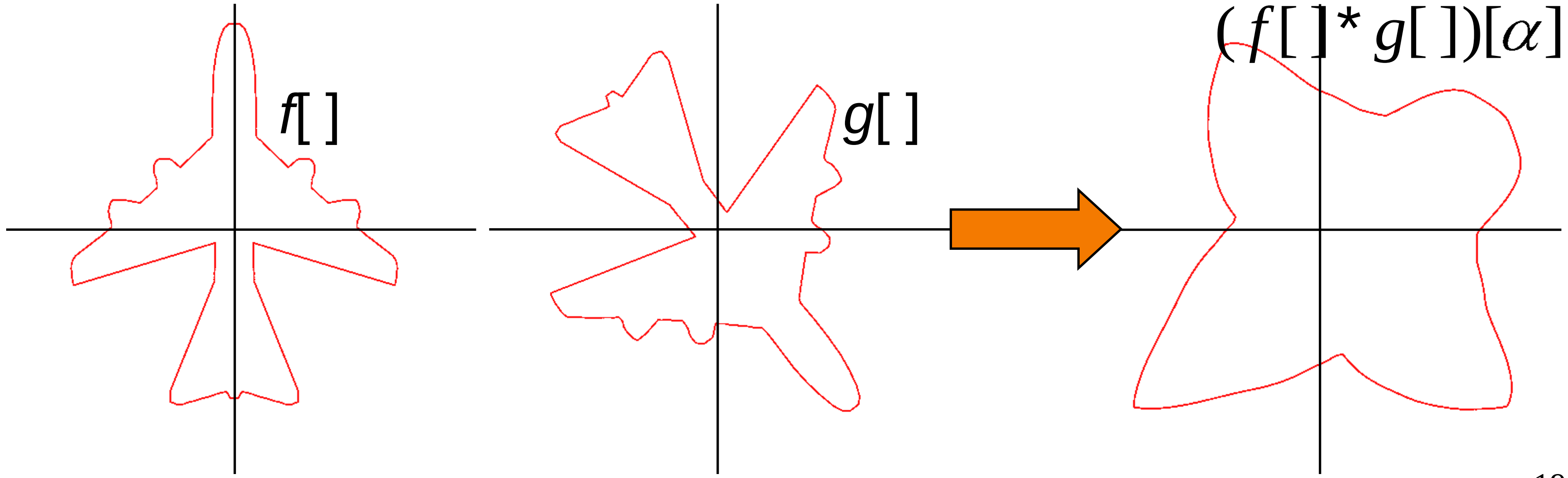
$$\boxed{D^2\left(\rho_\alpha(f[\,]), g[\,]\right) = \|f[\,]\|^2 + \|g[\,]\|^2 - 2\langle \rho_\alpha(f[\,]), g[\,]\rangle}$$



Alignment

$$D^2\left(\rho_\alpha(f[\,]), g[\,]\right) = \|f[\,]\|^2 + \|g[\,]\|^2 - 2\boxed{\left\langle \rho_\alpha(f[\,]), g[\,]\right\rangle}$$

Instead of looking for the minimum of the moving L2-difference, why not just look for the maximum of the moving dot-product?



Outline

Alignment

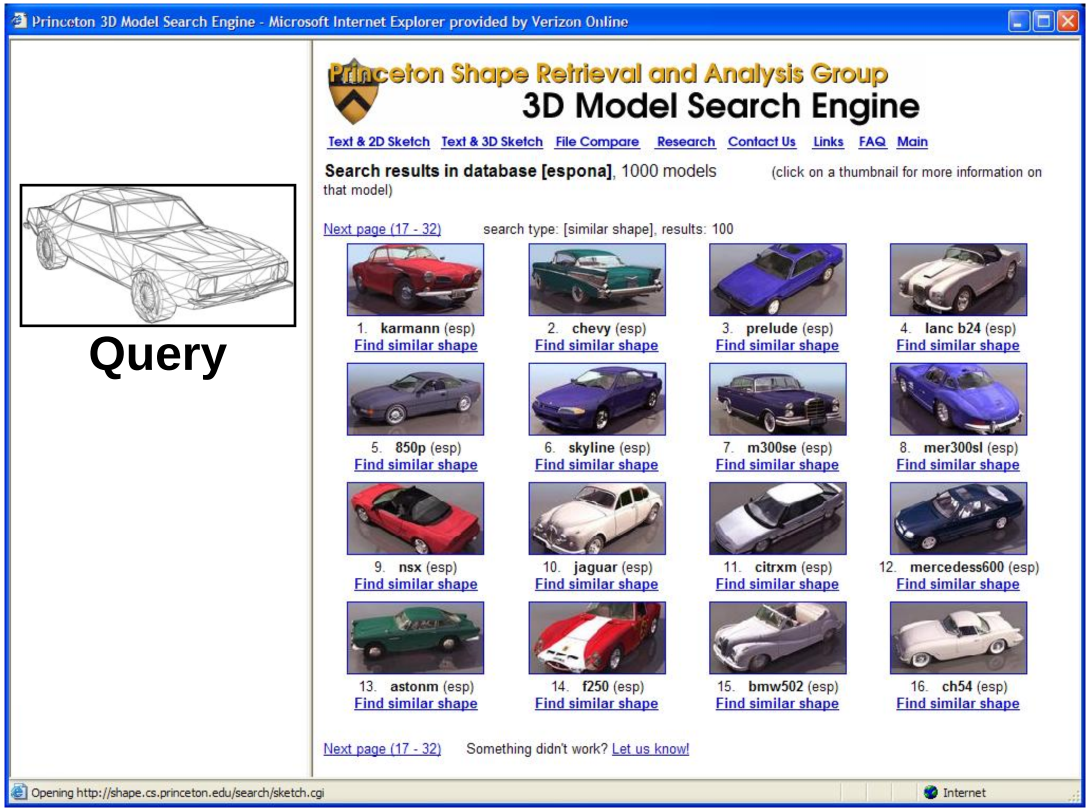
Shape Matching

Invariance

Pattern Matching

Shape Matching

In shape matching applications, we would like to find the shapes in a database that are most similar to a given query.


Princeton 3D Model Search Engine - Microsoft Internet Explorer provided by Verizon Online

Query

Princeton Shape Retrieval and Analysis Group
3D Model Search Engine

Text & 2D Sketch | Text & 3D Sketch | File Compare | Research | Contact Us | Links | FAQ | Main

Search results in database [espona], 1000 models (click on a thumbnail for more information on that model)

Next page (17 - 32) search type: [similar shape], results: 100

1. **karmann** (esp) Find similar shape	2. **chevy** (esp) Find similar shape	3. **prelude** (esp) Find similar shape	4. **lanc b24** (esp) Find similar shape
5. **850p** (esp) Find similar shape	6. **skyline** (esp) Find similar shape	7. **m300se** (esp) Find similar shape	8. **mer300sl** (esp) Find similar shape
9. **nsx** (esp) Find similar shape	10. **jaguar** (esp) Find similar shape	11. **citrxm** (esp) Find similar shape	12. **mercedess600** (esp) Find similar shape
13. **astonm** (esp) Find similar shape	14. **f250** (esp) Find similar shape	15. **bmw502** (esp) Find similar shape	16. **ch54** (esp) Find similar shape

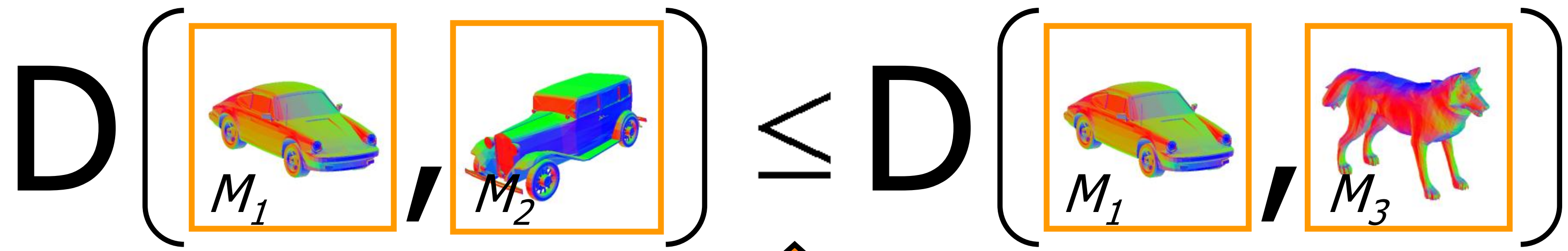
Next page (17 - 32) Something didn't work? Let us know!

Opening http://shape.cs.princeton.edu/search/sketch.cgi Internet

Shape Matching

General approach:

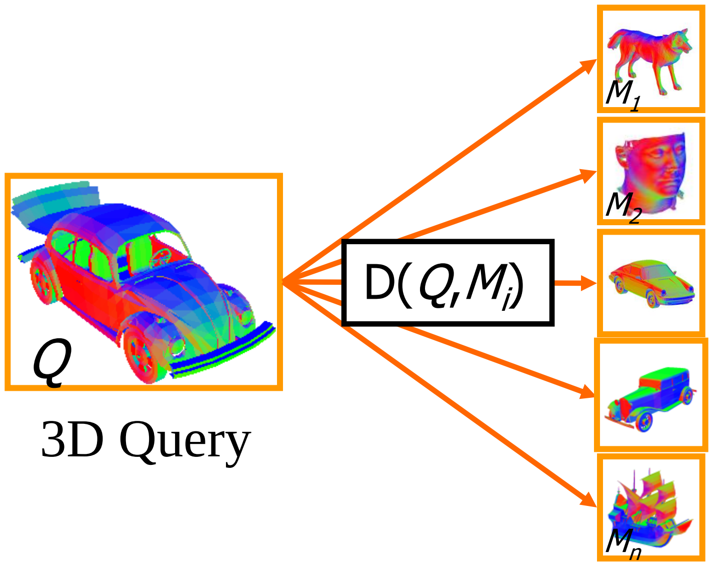
Define a function that takes in two models and returns a measure of their proximity.



$\boxed{M_1 \text{ is closer to } M_2 \text{ than it is to } M_3}$

Database Retrieval

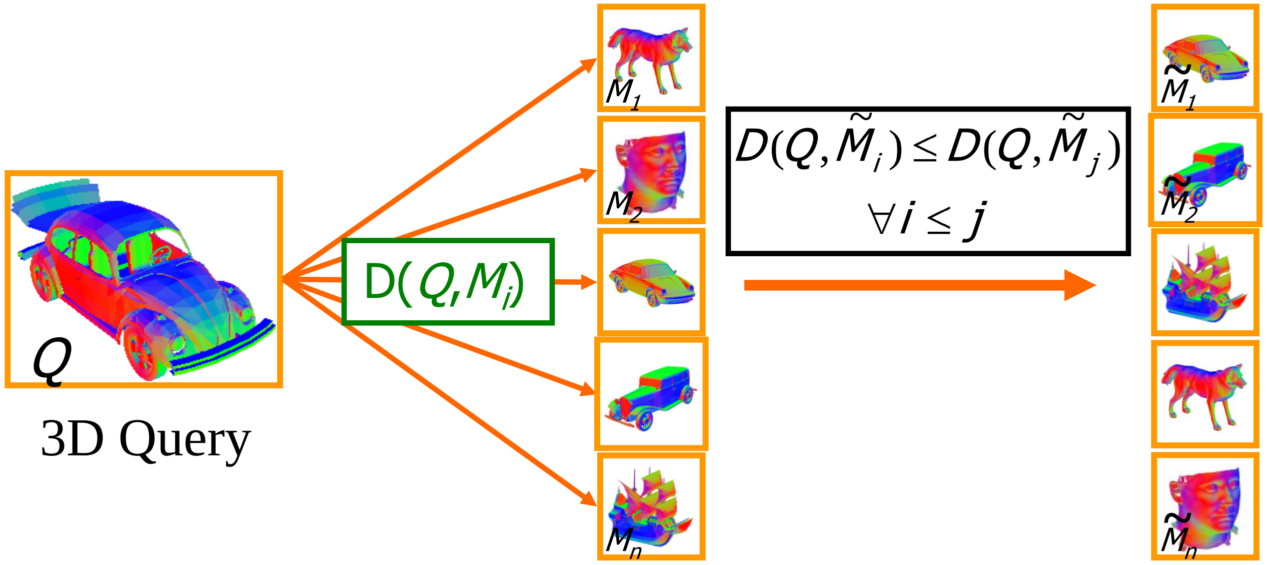
- Compute the distance from the query to each database model



Database Models

Database Retrieval

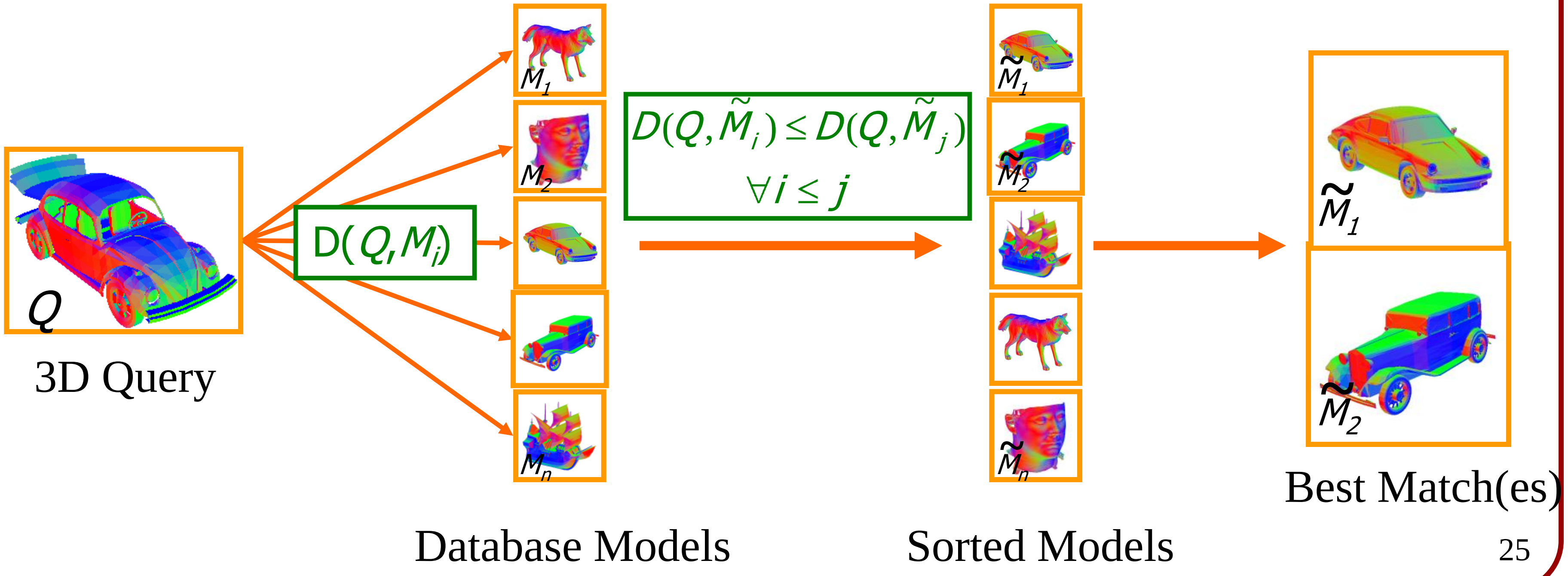
- Sort the database models by proximity



Database Models   Sorted Models

Database Retrieval

- Return the closest matches



Shape Matching

In shape matching applications, we would like to find the shapes in a database that are most similar to a given query.

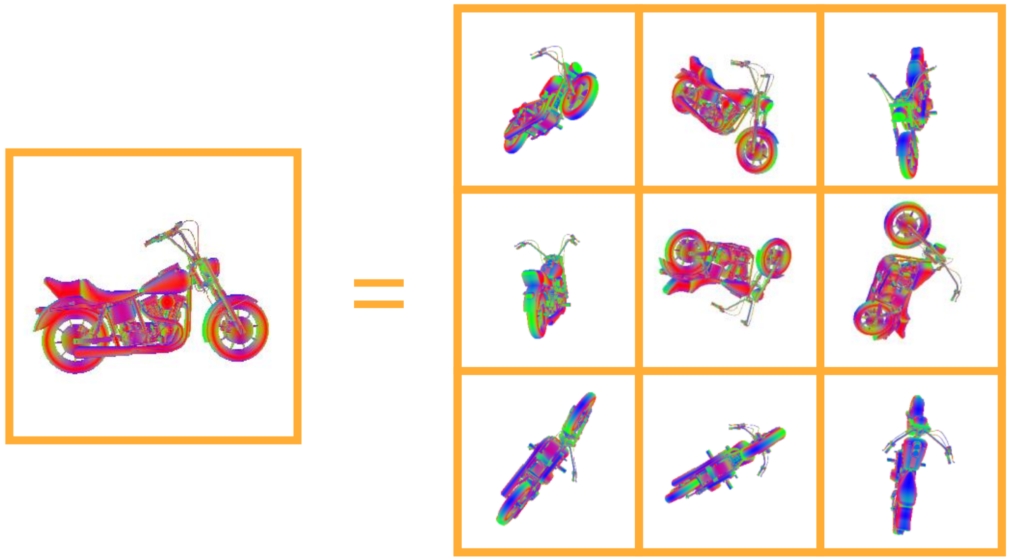
To do this efficiently, models are often represented by *Shape Descriptors*:

- Arrays of values encapsulating information about the shape of the model, such that
- The distance between the arrays gives a measure of proximity of the underlying shapes.

Shape Matching

Challenge:

Since the shape of the model doesn't change if we rotate it, we would like to match models across rotational poses.



Shape Matching

Challenge:

Since the shape of the model doesn't change if we rotate it, we would like to match models across rotational poses.

Solution 1:

One way to resolve is this is to define the measure of similarity by computing the cross-correlation, to find the distance between two models at the best possible alignment.

Shape Matching

Challenge:

Since the shape of the model doesn't change if we rotate it, we would like to match models across rotational poses.

Solu

One

measure of similarity by computing the cross-correlation, to find the distance between two models at the best possible alignment.

This can be too slow for interactive applications that need to return the best match from very large databases.

Shape Matching

Challenge:

Since the shape of the model doesn't change if we rotate it, we would like to match models across rotational poses.

Solu

One

measure of similarity by computing the cross-

corr

mod

This can be too slow for interactive applications that need to return the best match from very large databases.

Not quite true for 1D arrays, but becomes more true as the dimension increases.

Shape Matching

Challenge:

Since the shape of the model doesn’t change if we rotate it, we would like to match models across rotational poses.

Solution 2:

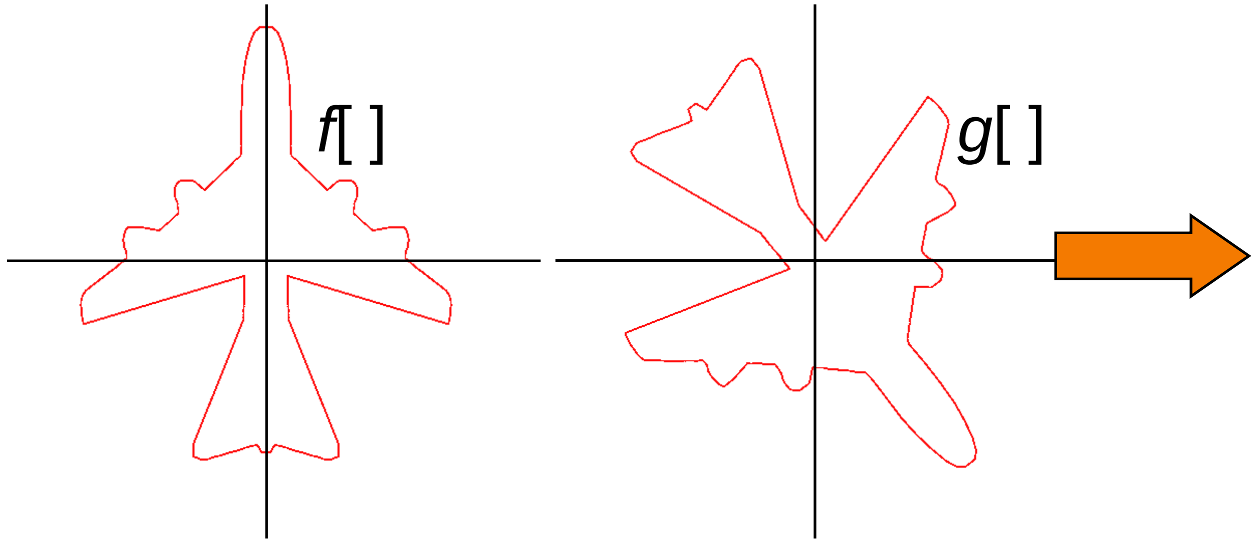
Alternatively, we can try to design a shape descriptor that is rotation invariant:

- Instances of the same shape in the same pose will give the same shape descriptor.

Invariance

Given an array *f*[ ], we would like to define a (possibly non-linear) mapping Φ taking *f*[ ] to some array *g*[ ], such that for all α:

$$\Phi(f[\,]) = \Phi\left(\rho_\alpha(f[\,])\right)$$



$$\Phi(f[\,]) = \Phi(g[\,])$$

Invariance

Given an array *f*[ ], expressed in terms of its Fourier decomposition:

$$f[\,] = \sum_{k=0}^{n-1} \hat{f}[k] v_k[\,]$$

What happens to the Fourier coefficients of *f*[ ] as we rotate *f*[ ]?

Invariance

Given an array *f*[ ], expressed in terms of its Fourier decomposition:

$$f[\,] = \sum_{k=0}^{n-1} \hat{f}[k] v_k[\,]$$

What happens to the Fourier coefficients of *f*[ ] as we rotate *f*[ ]?

$$\rho_\alpha(f[\,]) = \sum_{k=0}^{n-1} \hat{f}[k] \rho_\alpha(v_k[\,])$$

Invariance

$$\boxed{\rho_\alpha(f[\,]) = \sum_{k=0}^{n-1} \hat{f}[k]\rho_\alpha(v_k[\,])}$$

Since the $v_k[\,]$are a basis for the one-dimensional irreducible representations of the rotation group, we know that:

$$\rho_\alpha(v_k[\,]) = \lambda_k(\alpha)v_k[\,]$$

where $\lambda_k(\alpha)$ is some complex number.

Invariance

$$\boxed{\rho_\alpha(f[\,]) = \sum_{k=0}^{n-1} \hat{f}[k]\rho_\alpha(v_k[\,])}$$

Since the $v_k[\,]$are a basis for the one-dimensional irreducible representations of the rotation group, we know that:

$$\rho_\alpha(v_k[\,]) = \lambda_k(\alpha)v_k[\,]$$

where $\lambda_k(\alpha)$ is some complex number.

Since the representation is unitary, we know that $\lambda_k(\alpha)$ must have unit complex norm.

Invariance

$$\boxed{\rho_\alpha(f[\,])=\sum_{k=0}^{n-1}\hat{f}[k]\rho_\alpha(v_k[\,])}$$

Thus, if:

$$\hat{f}[0],\hat{f}[1],\ldots,\hat{f}[n-1]$$

Are the Fourier coefficients of $f[\,]$, then the Fourier coefficients of $\rho_\alpha(f[\,])$ will be:

$$\lambda_0(\alpha)\hat{f}[0],\lambda_1(\alpha)\hat{f}[1],\ldots,\lambda_{n-1}(\alpha)\hat{f}[n-1]$$

Invariance

$$\boxed{\rho_\alpha(f[\,]) = \sum_{k=0}^{n-1} \hat{f}[k]\rho_\alpha(v_k[\,])}$$

Thus, if:

$$\hat{f}[0], \hat{f}[1], \ldots, \hat{f}[n-1]$$

Are the Fourier coefficients of *f*[ ], then the Fourier coefficients of ρ_α(*f*[ ]) will be:

$$\lambda_0(\alpha)\hat{f}[0], \lambda_1(\alpha)\hat{f}[1], \ldots, \lambda_{n-1}(\alpha)\hat{f}[n-1]$$

and we have:

$$\left\|\lambda_k(\alpha)\hat{f}[k]\right\| = \left\|\hat{f}[k]\right\|$$

for all *k*.

Invariance

$$\boxed{\left\| \lambda_k(\alpha)\,\hat{f}[k] \right\| = \left\| \hat{f}[k] \right\|}$$

So, we can get a rotation invariant representation of *f*[ ] by storing only the magnitudes of the Fourier coefficients (i.e. discarding phase):

$$\Phi(f[\,]) = \left\|\hat{f}[0]\right\|, \left\|\hat{f}[1]\right\|, \ldots, \left\|\hat{f}[n-1]\right\|$$

Invariance

What kind of information do we get when we compare just the amplitudes of the Fourier coefficients?

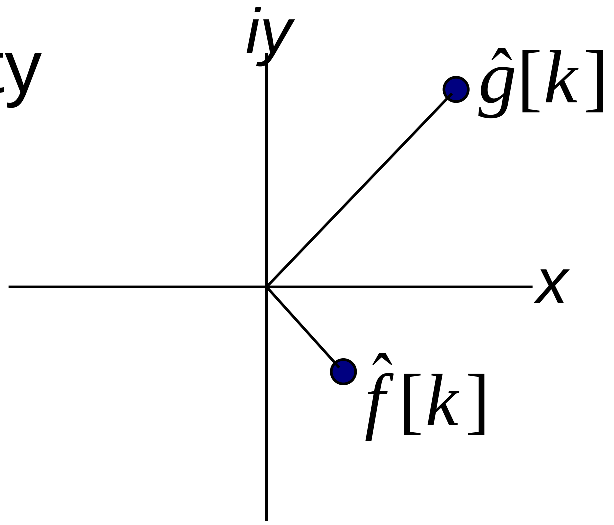
Invariance

Suppose that we are given two arrays *f*[ ] and *g*[ ] with only one non-zero Fourier coefficient:

$$f[\,] = \hat{f}[k]v_k[\,]$$

$$g[\,] = \hat{g}[k]v_k[\,]$$

what is the measure of similarity at the optimal alignment?



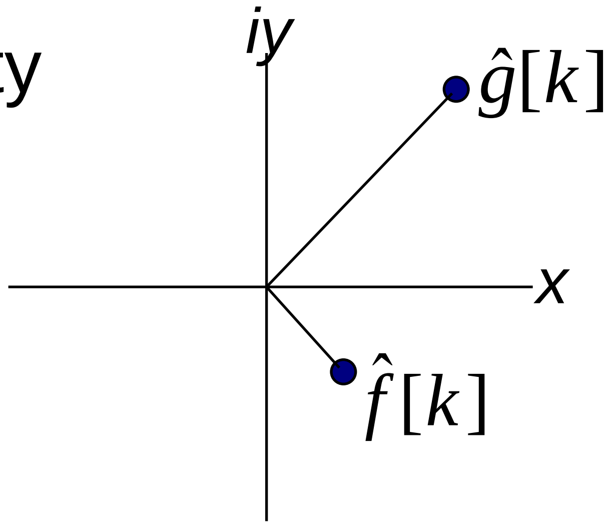
Invariance

Suppose that we are given two arrays *f*[ ] and *g*[ ] with only one non-zero Fourier coefficient:

$$f[\,] = \hat{f}[k]v_k[\,]$$
$$g[\,] = \hat{g}[k]v_k[\,]$$

what is the measure of similarity at the optimal alignment?

If we rotate *f*[ ] by α, this amounts to multiplying the *k*-th Fourier coefficient by $e^{-ik\alpha}$.



Invariance

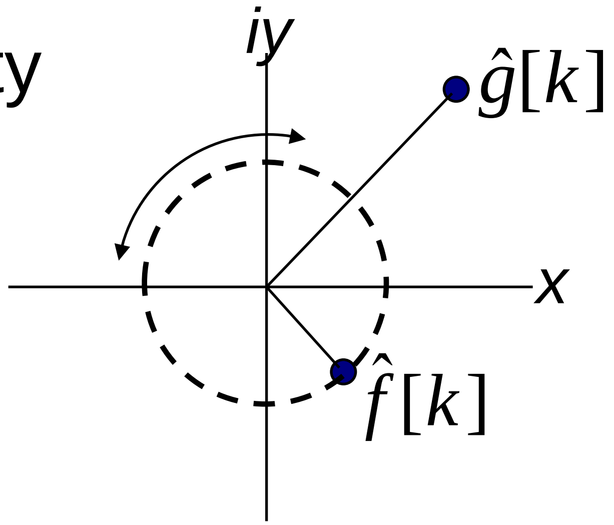
Suppose that we are given two arrays *f*[ ] and *g*[ ] with only one non-zero Fourier coefficient:

$$f[\,] = \hat{f}[k]v_k[\,]$$

$$g[\,] = \hat{g}[k]v_k[\,]$$

what is the measure of similarity at the optimal alignment?

If we rotate *f*[ ] by α, this amounts to multiplying the *k*-th Fourier coefficient by $e^{-ik\alpha}$.



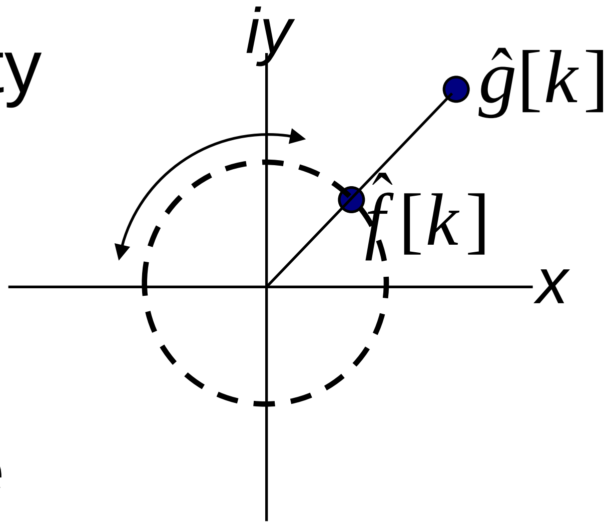
But this is just a rotation in the complex plane.

Invariance

Suppose that we are given two arrays *f*[ ] and *g*[ ] with only one non-zero Fourier coefficient:

$$f[\,] = \hat{f}[k]v_k[\,]$$
$$g[\,] = \hat{g}[k]v_k[\,]$$

what is the measure of similarity at the optimal alignment?



At the optimal rotation, the Fourier coefficients are on the same line and the measure of similarity is the difference between the lengths.

Invariance

Thus, storing only the amplitude of the Fourier coefficients and discarding phase, we get a shape representation $\Phi(f[\ ])$ that:

- Is invariant to rotations
- Provides a measure of similarity that corresponds to the distance between *f*[ ] and *g*[ ] if we could optimally align the different frequency components independently.

Invariance

Thus, storing only the amplitude of the Fourier coefficients and discarding phase, we get a shape representation $\Phi(f[\,])$ that:

- Is invariant to rotations
- Provides a measure of similarity that corresponds to the distance between *f*[ ] and *g*[ ] if we could optimally align the different frequency components independently.

$$\Downarrow$$

- This is a lower bound for the distance between *f*[ ] and *g*[ ] at the optimal alignment.

Invariance

How good is the lower bound?

After discarding phase, what’s left?

Invariance

How good is the lower bound?

After discarding phase, what’s left?

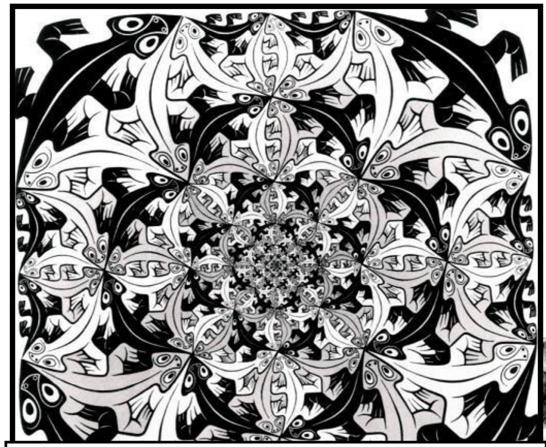
<u>Experiment</u>:

To test this, we can consider what happens when we take two arrays and swap the amplitudes of the Fourier coefficients:

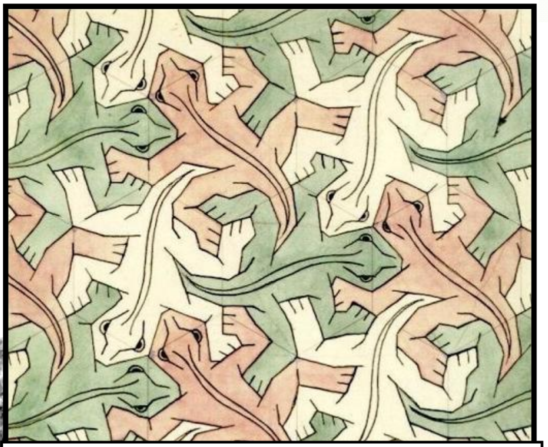
$$f[\,] = \sum_{k=0}^{n-1} \boxed{r_k} e^{i\theta_k} v_k[\,] \qquad\qquad g[\,] = \sum_{k=0}^{n-1} s_k e^{\boxed{i\phi_k}} v_k[\,]$$

$$\text{ASwap}(f[\,], g[\,]) = \sum_{k=0}^{n-1} \boxed{r_k} e^{\boxed{i\phi_k}} v_k[\,]$$

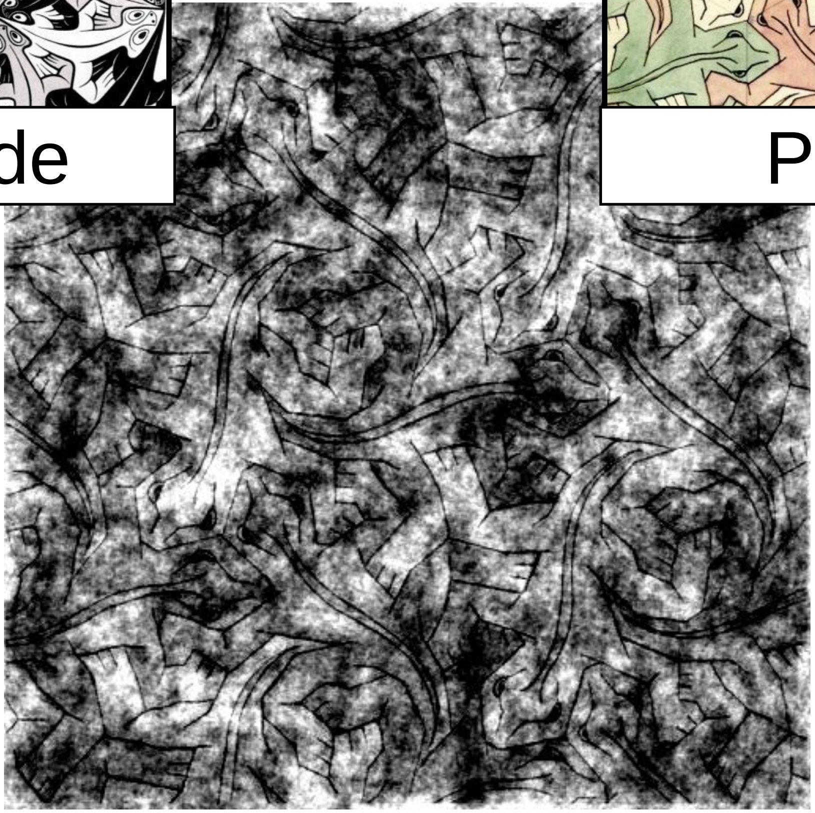
Invariance



Amplitude



Phase



Invariance

Does this imply that most of the information is stored in the phase?

Invariance

Does this imply that most of the information is stored in the phase?

Not necessarily. For human perception, dominant information occurs at image boundaries:

- Positions in the image where there is an abrupt change of value

Invariance

Does this imply that most of the information is stored in the phase?

Not necessarily. For human perception, dominant information occurs at image boundaries:

- Positions in the image where there is an abrupt change of value

These discontinuities arise when the phases are lined up so the occurrence of these events is strongly phase dependent.

Invariance

Does this imply that most of the information is stored in the phase?

Not necessarily. For human perception, dominant information occurs at image boundaries:

- Positions in the image where there is an abrupt change of value

These discontinuities arise when the phases are lined up so the occurrence of these events is strongly phase dependent.

If the grid encodes other type of information (non-visual) amplitude can be more important.

Outline

Alignment

Shape Matching

Invariance

Pattern Matching

Notation

If *f*[ ] and *g*[ ] are two *n*-dimensional arrays, then we can define *f*[ ]·*g*[ ] to be the entry-wise product of the two arrays, with:

$$\left(f[\,]\cdot g[\,]\right)[j] = f[j]\cdot g[j]$$

Note 1

If *f*[ ] is a real-valued *n*-dimensional array, and *g*[ ] and *h*[ ] are complex-valued *n*-dimensional arrays, then:

$$\left\langle f[\,]\cdot g[\,], h[\,]\right\rangle = \left\langle g[\,], f[\,]\cdot h[\,]\right\rangle$$

Note 1

If *f*[ ] is a real-valued *n*-dimensional array, and *g*[ ] and *h*[ ] are complex-valued *n*-dimensional arrays, then:

$$\left\langle f[\,]\cdot g[\,], h[\,]\right\rangle = \sum_{k=0}^{n-1} \left(f[\,]\cdot g[\,]\right)[k]\cdot \overline{h[k]}$$

Note 1

If *f*[ ] is a real-valued *n*-dimensional array, and *g*[ ] and *h*[ ] are complex-valued *n*-dimensional arrays, then:

$$\begin{aligned}\left\langle f[\,]\cdot g[\,], h[\,]\right\rangle &= \sum_{k=0}^{n-1} \left(f[\,]\cdot g[\,]\right)[k]\cdot\overline{h[k]}\\ &= \sum_{k=0}^{n-1} f[k]\cdot g[k]\cdot\overline{h[k]}\end{aligned}$$

Note 1

If *f*[ ] is a real-valued *n*-dimensional array, and *g*[ ] and *h*[ ] are complex-valued *n*-dimensional arrays, then:

$$\left\langle f[\,]\cdot g[\,], h[\,]\right\rangle = \sum_{k=0}^{n-1} \left(f[\,]\cdot g[\,]\right)[k]\cdot\overline{h[k]}$$

$$= \sum_{k=0}^{n-1} f[k]\cdot g[k]\cdot\overline{h[k]}$$

$$= \sum_{k=0}^{n-1} g[k]\cdot\overline{f[k]}\cdot\overline{h[k]}$$

Note 1

If *f*[ ] is a real-valued *n*-dimensional array, and *g*[ ] and *h*[ ] are complex-valued *n*-dimensional arrays, then:

$$
\begin{aligned}
\left\langle f[\,]\cdot g[\,], h[\,]\right\rangle &= \sum_{k=0}^{n-1} \left(f[\,]\cdot g[\,]\right)[k]\cdot\overline{h[k]} \\
&= \sum_{k=0}^{n-1} f[k]\cdot g[k]\cdot\overline{h[k]} \\
&= \sum_{k=0}^{n-1} g[k]\cdot\overline{f[k]}\cdot\overline{h[k]} \\
&= \sum_{k=0}^{n-1} g[k]\cdot\overline{\left(f[\,]\cdot h[\,]\right)[k]}
\end{aligned}
$$

Note 1

If *f*[ ] is a real-valued *n*-dimensional array, and *g*[ ] and *h*[ ] are complex-valued *n*-dimensional arrays, then:

$$
\begin{aligned}
\langle f[\,]\cdot g[\,], h[\,]\rangle &= \sum_{k=0}^{n-1} \left(f[\,]\cdot g[\,]\right)[k]\cdot \overline{h[k]} \\
&= \sum_{k=0}^{n-1} f[k]\cdot g[k]\cdot \overline{h[k]} \\
&= \sum_{k=0}^{n-1} g[k]\cdot \overline{f[k]}\cdot \overline{h[k]} \\
&= \sum_{k=0}^{n-1} g[k]\cdot \overline{\left(f[\,]\cdot h[\,]\right)[k]} \\
&= \langle g[\,], f[\,]\cdot h[\,]\rangle
\end{aligned}
$$

Note 2

If ρ_α is the unitary representation that shifts an array by α indices:

$$\rho_\alpha(f[\,])[k] = f[k-\alpha]$$

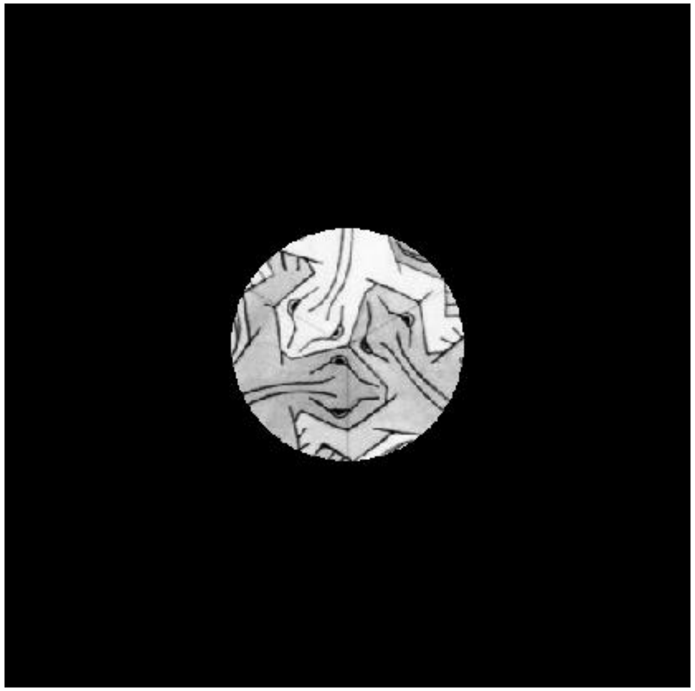
Then

$$\left(\rho_\alpha(f[\,]\cdot g[\,])\right) = \rho_\alpha(f[\,])\cdot\rho_\alpha(g[\,])$$

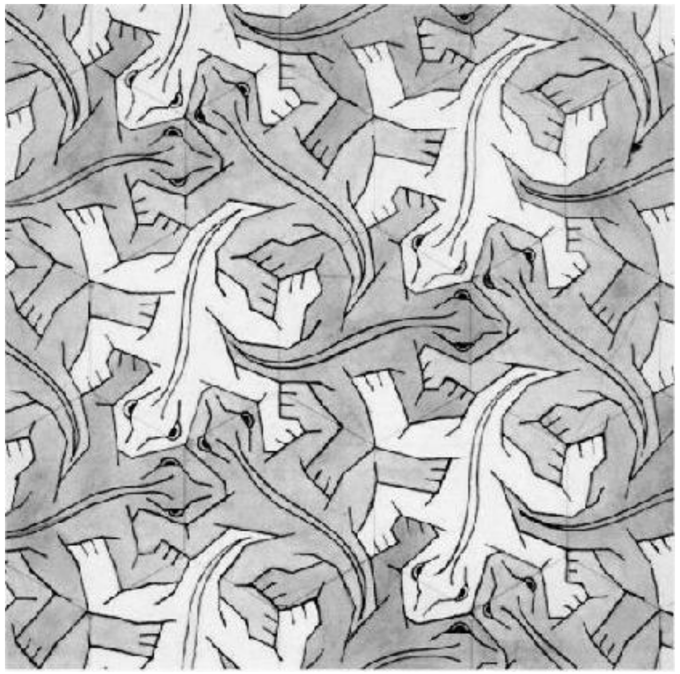
Pattern Matching

Given an instance of a pattern, find all occurrences of the pattern within a target image:

Pattern *f*[ ][ ]



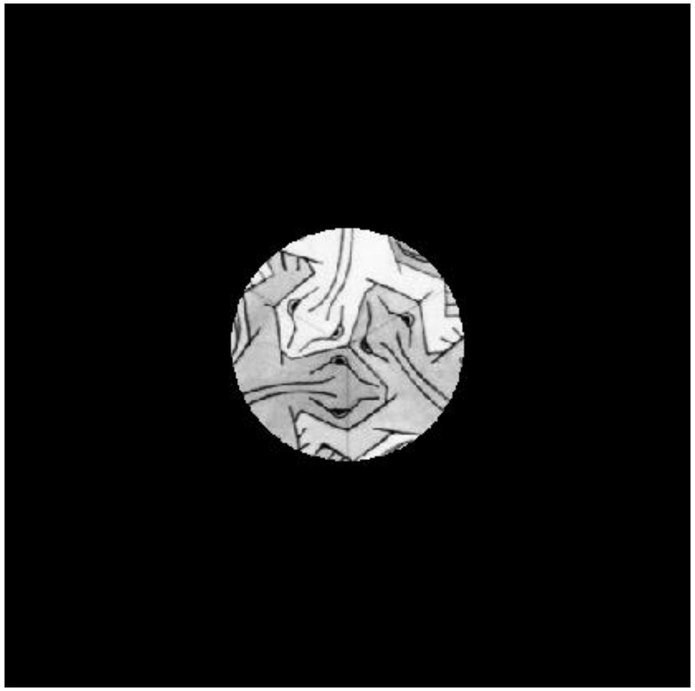
Target Image *g*[ ][ ]



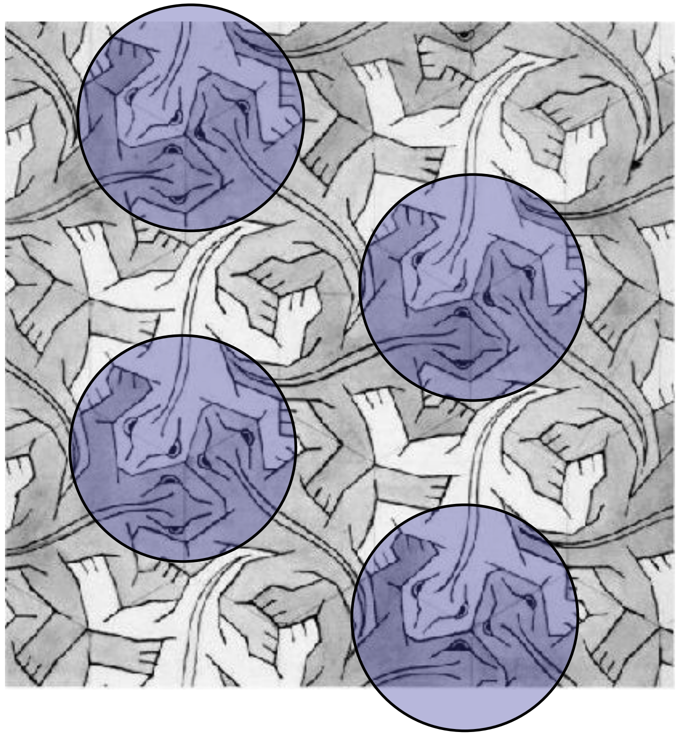
Pattern Matching

Given an instance of a pattern, find all occurrences of the pattern within a target image:

Pattern *f*[ ][ ]

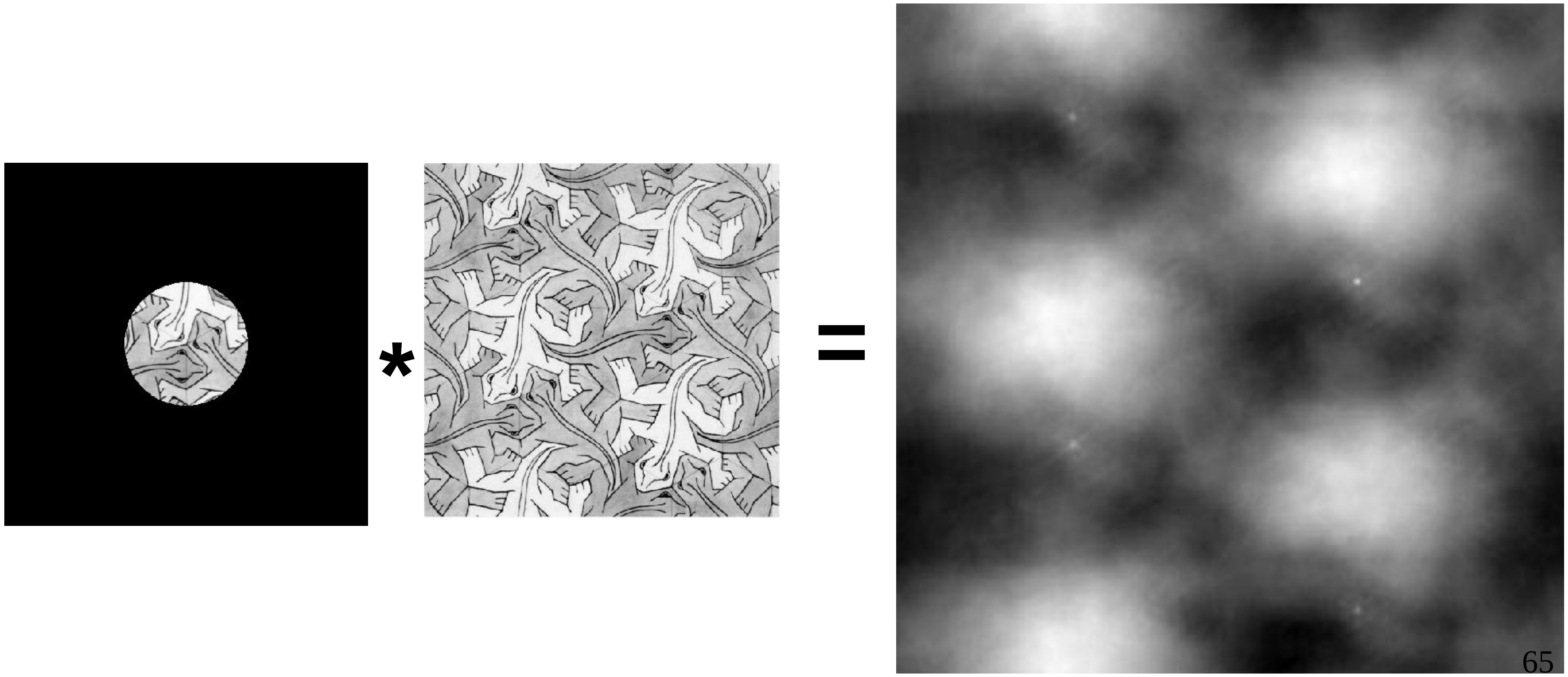


Target Image *g*[ ][ ]



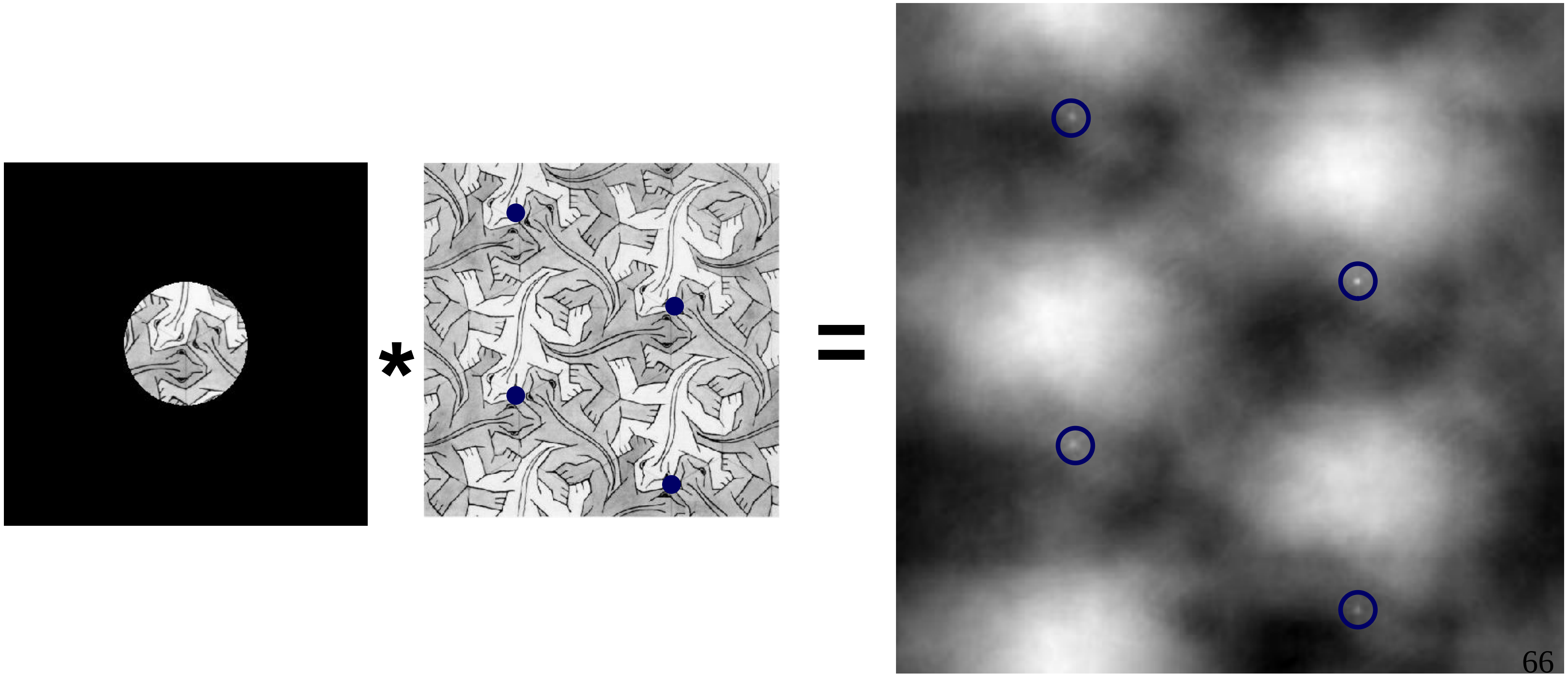
Pattern Matching

We could compute the cross correlation of the pattern with the image and look for local maxima:



Pattern Matching

We could compute the cross correlation of the pattern with the image and look for local maxima:



Pattern Matching

The cross-correlation has large values because the dot product of the image with the translated pattern instance is large.

What causes the dot-product of *f*[ ] with *g*[ ] to be large?

Pattern Matching

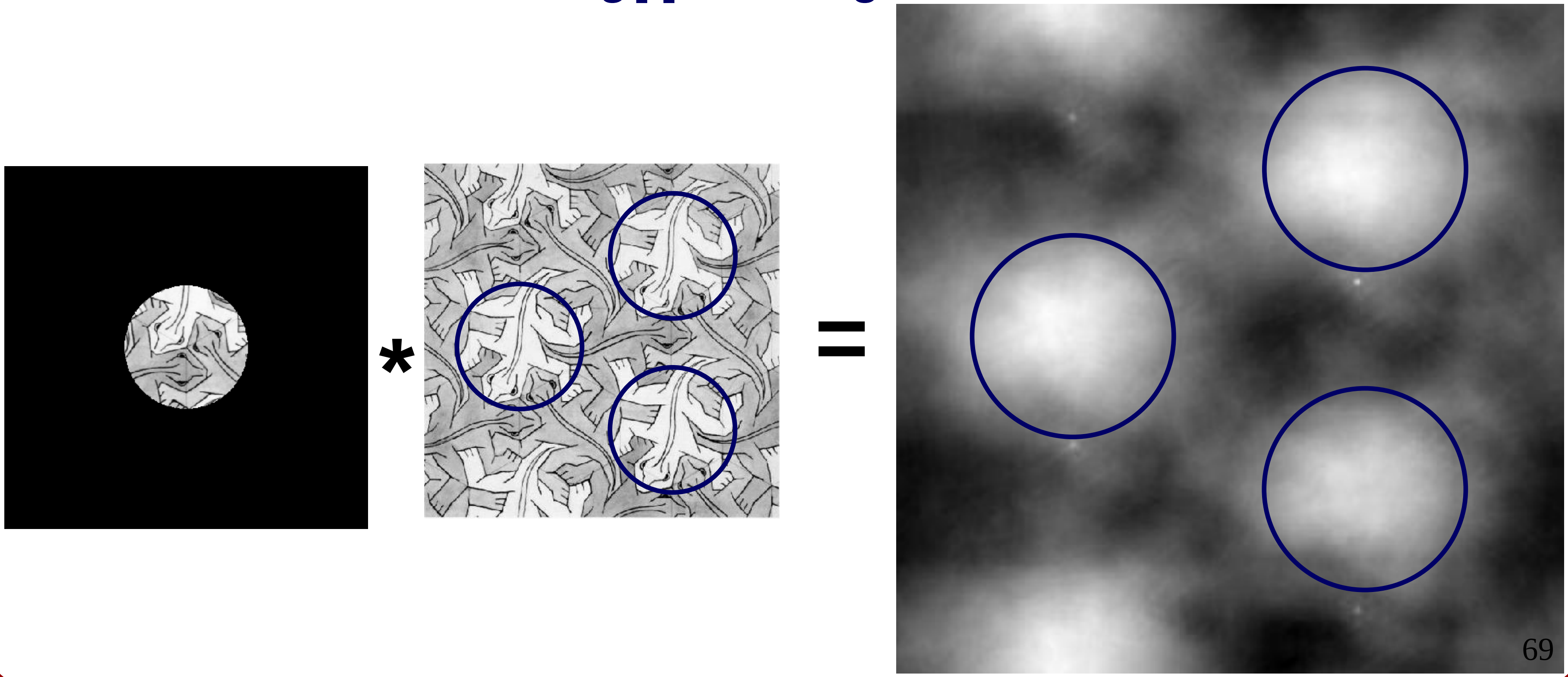
What causes the dot-product of *f*[ ] with *g*[ ] to be large?

- If the values of *f*[ ] are similar to the values of *g*[ ]

Pattern Matching

What causes the dot-product of *f*[ ] with *g*[ ] to be large?

- If the values of *f*[ ] are similar to the values of *g*[ ]
- If the values of *g*[ ] are large



Pattern Matching

We don't want to measure:

> How correlated is the pattern instance with a region in the image?

What we want to measure is:

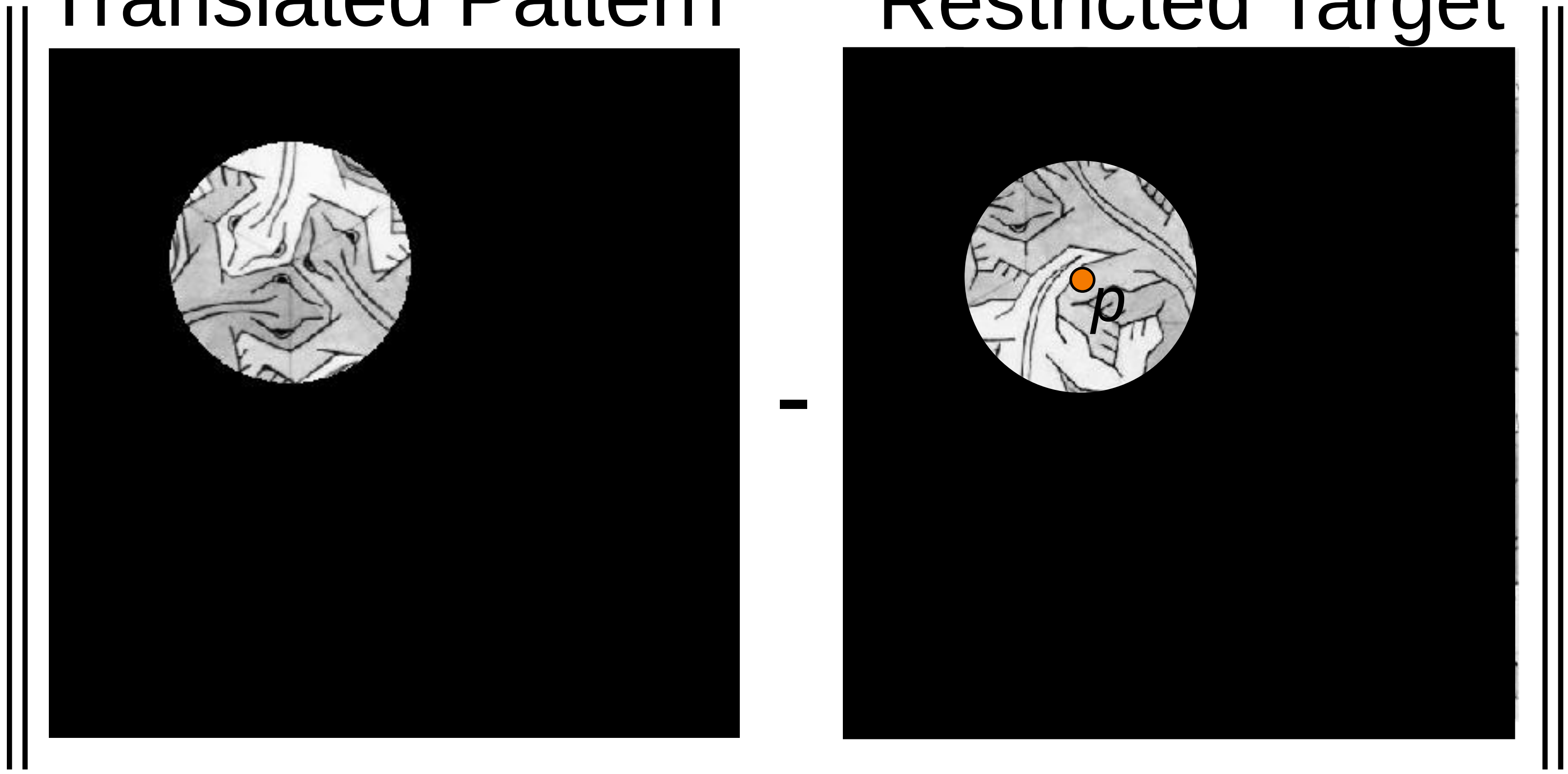
> How similar is the pattern instance with a region in the image?

Pattern Matching

For every point in the image, we want to know how similar the region about the point is to the translated pattern.

Translated Pattern

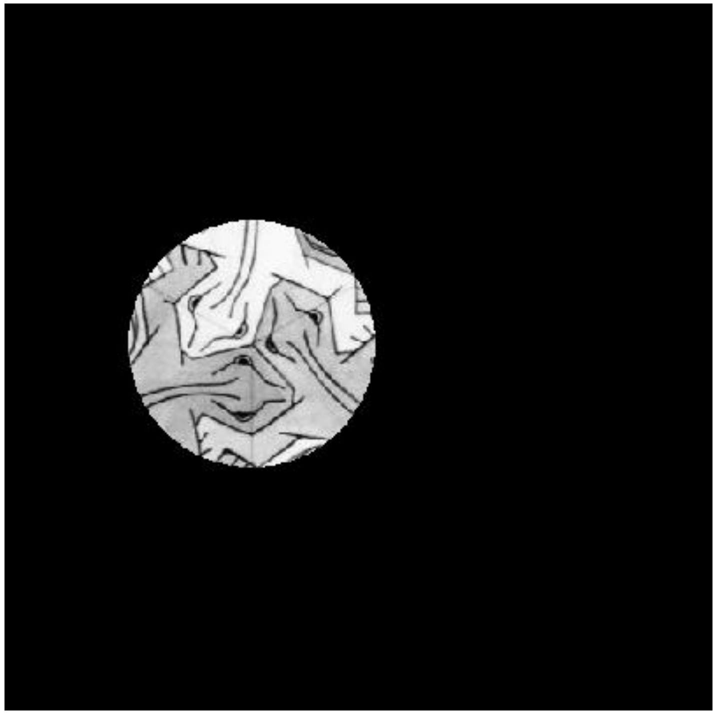
Restricted Target

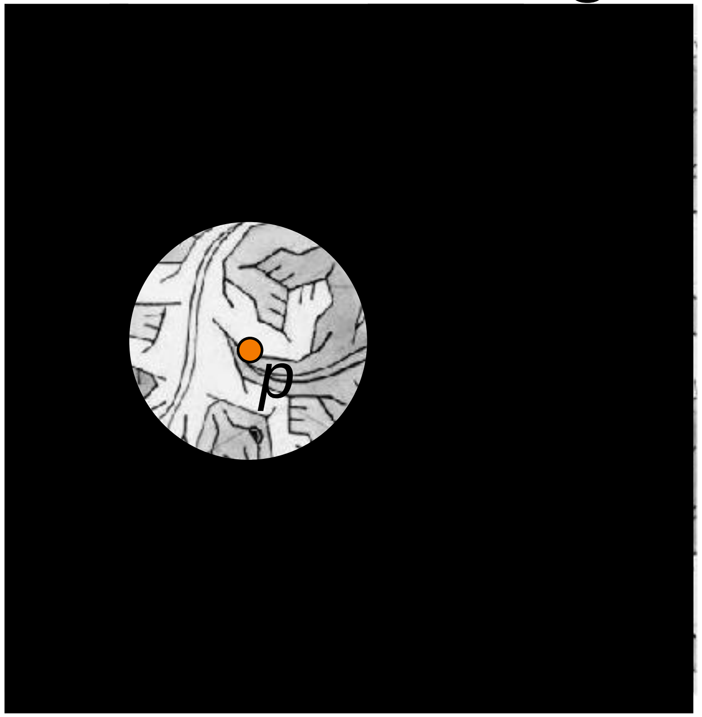


Pattern Matching

For every point in the image, we want to know how similar the region about the point is to the translated pattern.

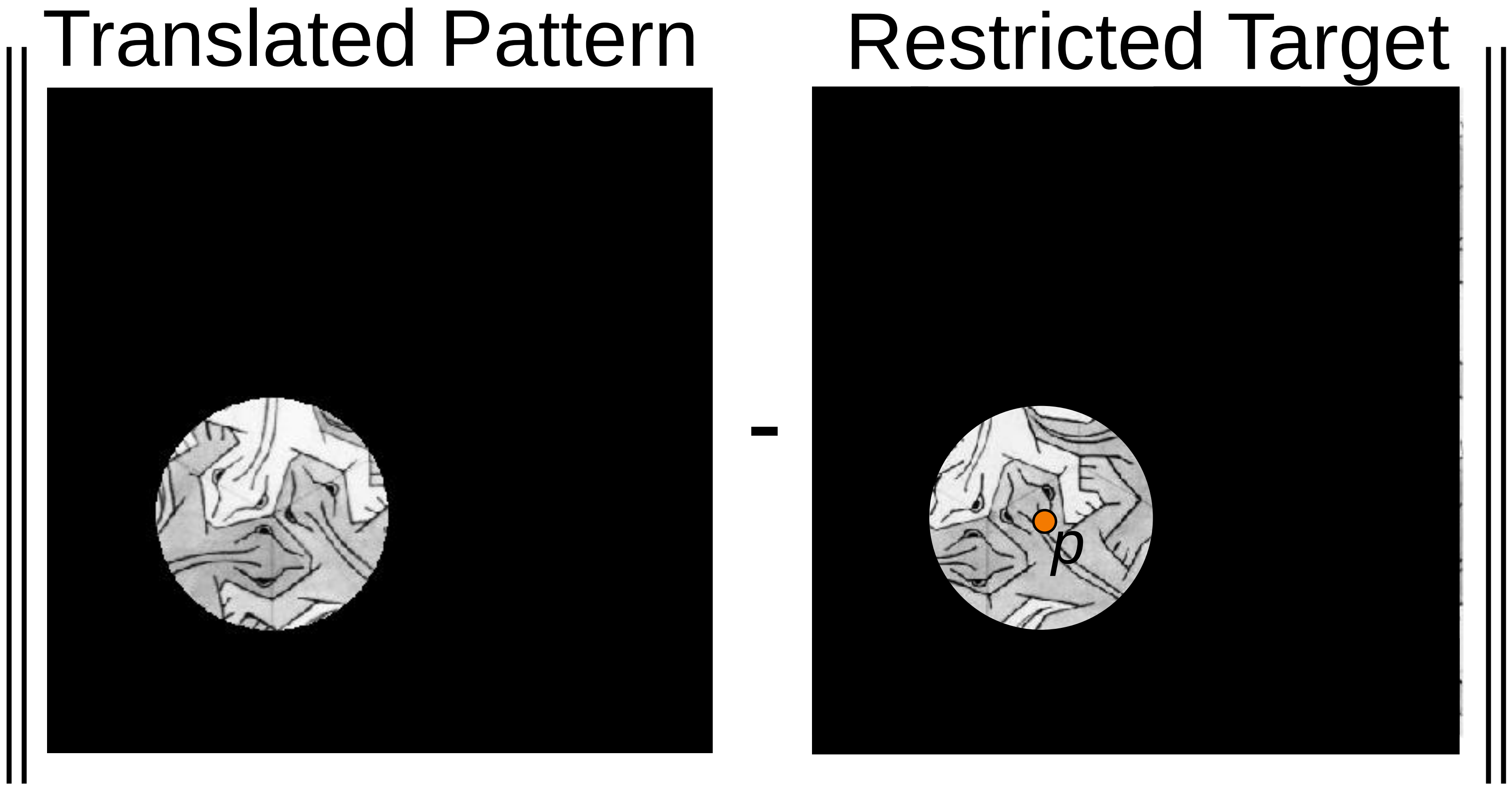
$$\left\| \text{Translated Pattern} - \text{Restricted Target} \right\|$$





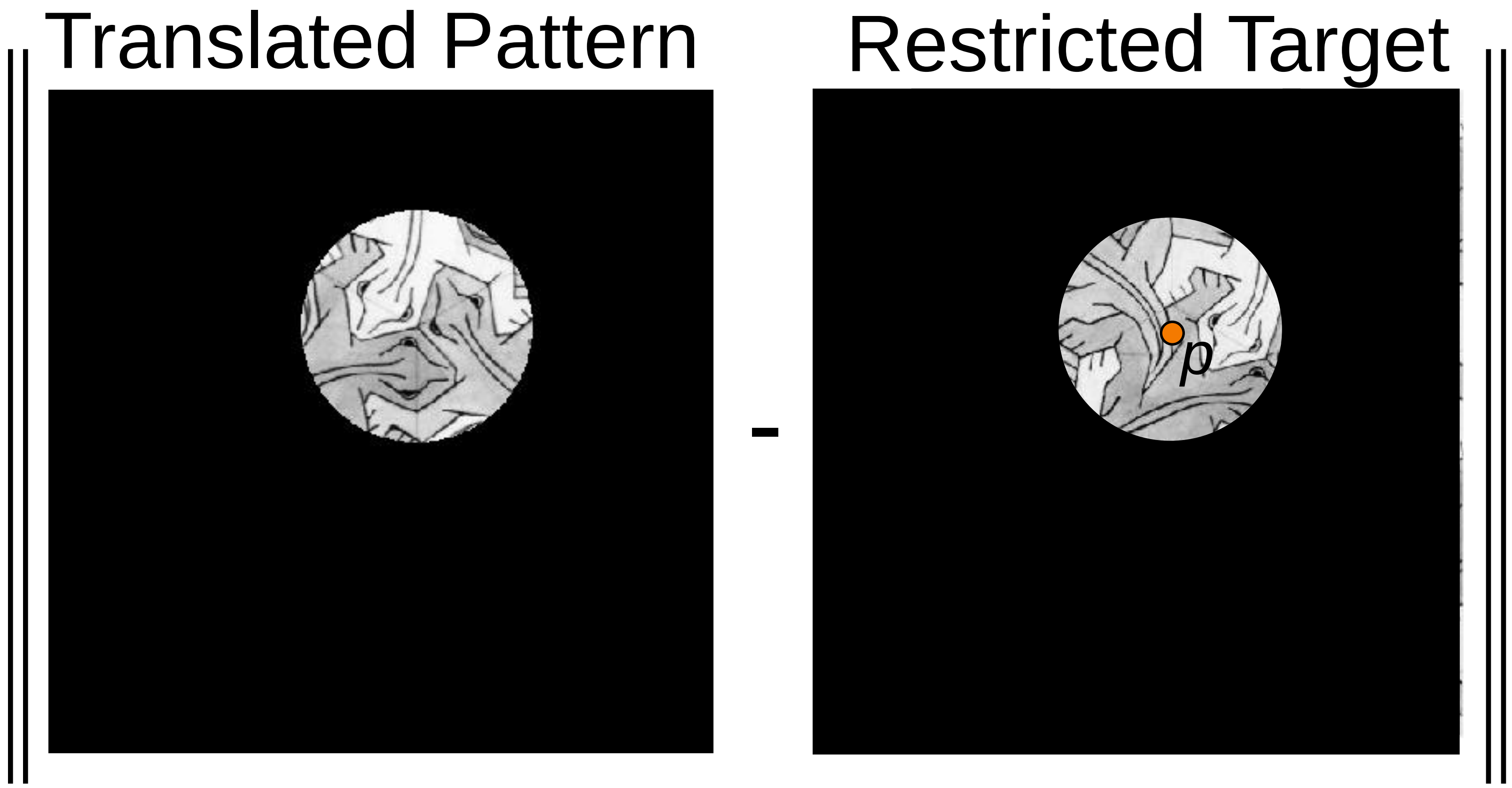
Pattern Matching

For every point in the image, we want to know how similar the region about the point is to the translated pattern.



Pattern Matching

For every point in the image, we want to know how similar the region about the point is to the translated pattern.



Pattern Matching

How do we express this formally?

Pattern Matching

How do we express this formally?

If we represent the pattern by *f*[ ][ ], then the translation of the pattern to the point *p* can be written as:

$$\rho_p(f[\,][\,])$$

Pattern Matching

How do we express this formally?

If we represent the pattern by *f*[ ][ ], then the translation of the pattern to the point *p* can be written as:

$$\rho_p(f[\ ][\ ])$$

How do we express the restriction of *g*[ ][ ] to the region about *p*?

Pattern Matching

How do we express this formally?

If we represent the pattern by *f*[ ][ ], then the translation of the pattern to the point *p* can be written as:

$$\rho_p(f[\,][\,])$$

How do we express the restriction of *g*[ ][ ] to the region about *p*?

What we want to do is to zero out all of *g*[ ][ ] except for the region about *p*.

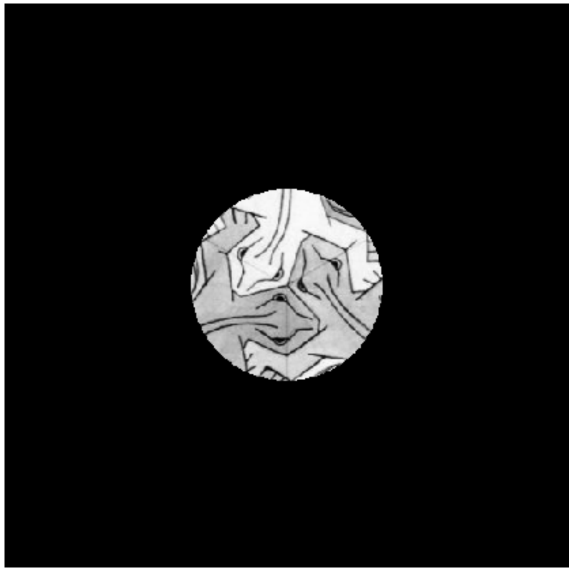
Pattern Matching

How do we express this formally?

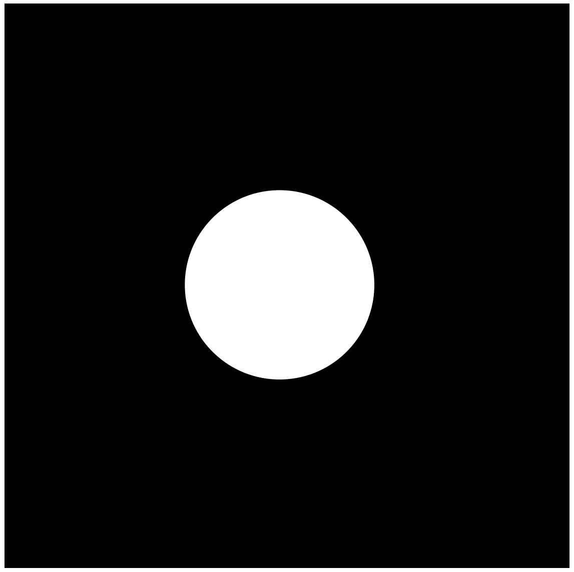
Let χ[ ][ ] be the characteristic grid of the pattern:

$$\chi[j][k] = \begin{cases} 1 & \text{if } (j,k) \text{ is inside the pattern} \\ 0 & \text{otherwise} \end{cases}$$

f[ ][ ]



χ[ ][ ]



Pattern Matching

How do we express this formally?

The restriction of *g*[ ][ ] to the region about *p* can be expressed as:

$$\rho_p(\chi[\,][\,])\cdot g[\,][\,]$$

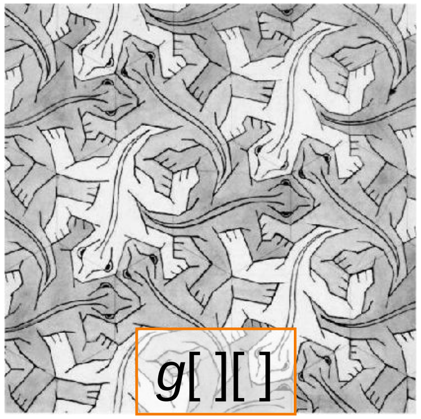
- $\rho_p(f[\,][\,])$ shifts the characteristic grid to be centered about *p*.
- Multiplying by $\rho_p(f[\,][\,])$ zeros out everything except for the region about *p*.

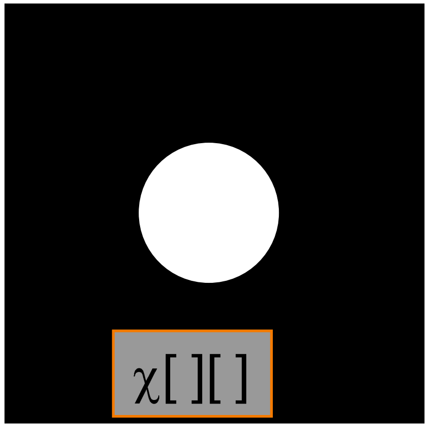
Pattern Matching

How do we express this formally?

The restriction of *g*[ ][ ] to the region about *p* can be expressed as:

$$\rho_p(\chi[\,][\,])\cdot g[\,][\,]$$



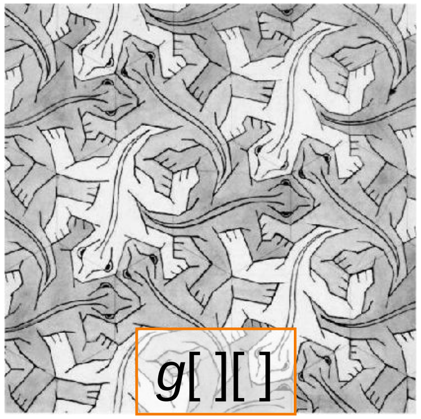


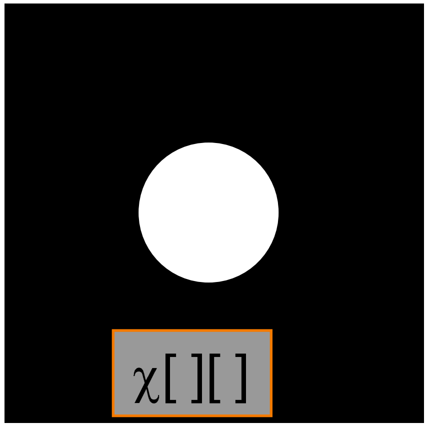
Pattern Matching

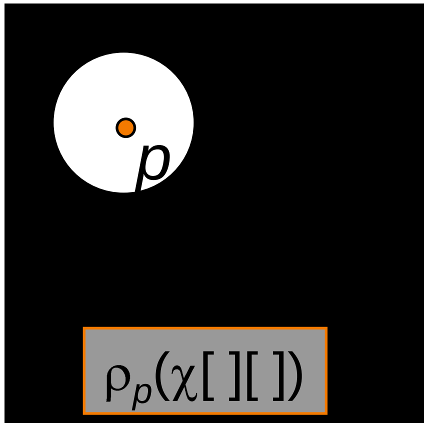
How do we express this formally?

The restriction of *g*[ ][ ] to the region about *p* can be expressed as:

$$\rho_p(\chi[\,][\,])\cdot g[\,][\,]$$





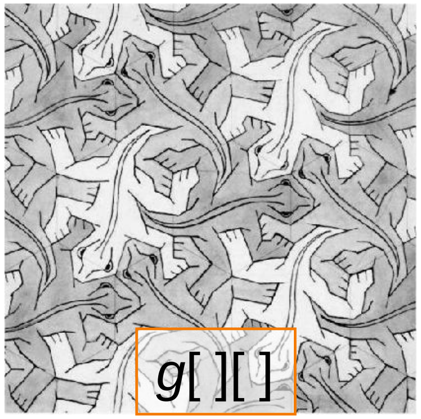


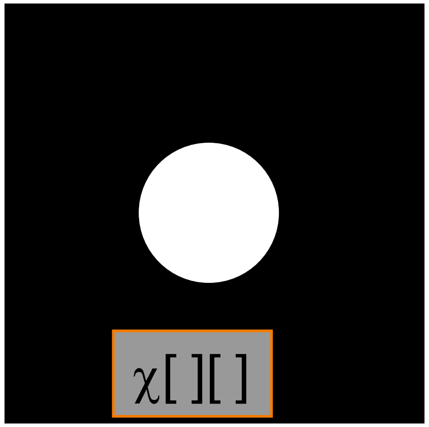
Pattern Matching

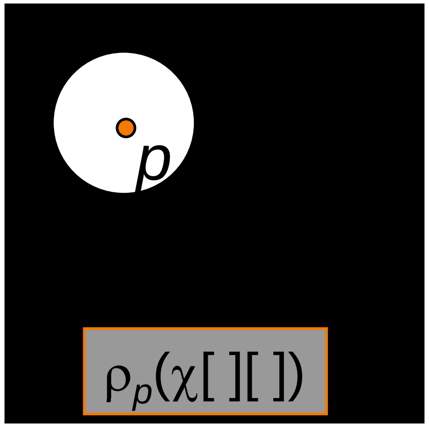
How do we express this formally?

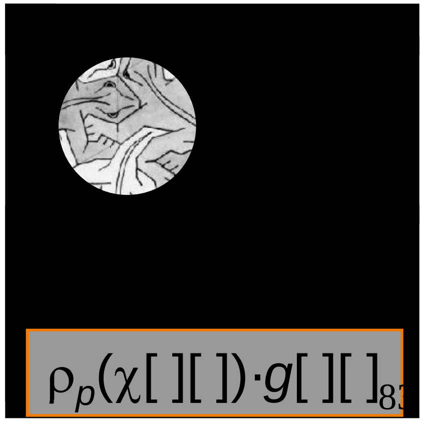
The restriction of *g*[ ][ ] to the region about *p* can be expressed as:

$$\rho_p(\chi[\,][\,])\cdot g[\,][\,]$$







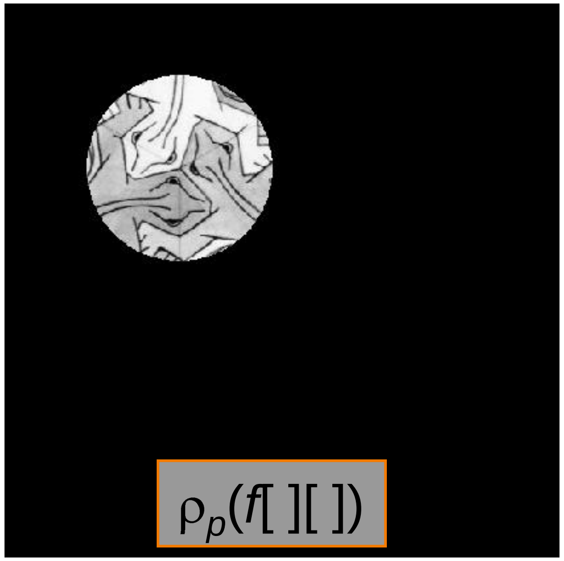


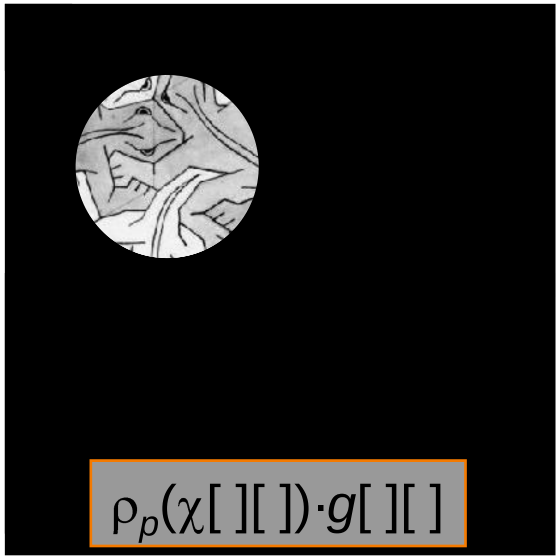
Pattern Matching

How do we express this formally?

For every *p*, we would like to compute:

$$\left\| \rho_p(f[\,][\,]) - \rho_p(\chi[\,][\,]) \cdot g[\,][\,] \right\|^2$$





Pattern Matching

How do we express this formally?

For every *p*, we would like to compute:

$$\left\| \rho_p(f[\,][\,]) - \rho_p(\chi[\,][\,]) \cdot g[\,][\,] \right\|^2$$

Writing this out in terms of dot-products gives three terms:

- $\left\langle \rho_p(f[\,][\,]), \rho_p(f[\,][\,]) \right\rangle$
- $-2\left\langle \rho_p(f[\,][\,]), \rho_p(\chi[\,][\,]) \cdot g[\,][\,] \right\rangle$
- $\left\langle \rho_p(\chi[\,][\,]) \cdot g[\,][\,], \rho_p(\chi[\,][\,]) \cdot g[\,][\,] \right\rangle$

Pattern Matching (Term 1)

$$\boxed{\left\langle \rho_p(f[\,][\,]), \rho_p(f[\,][\,]) \right\rangle}$$

Using the the fact that the representation is unitary gives:

$$\left\langle \rho_p(f[\,][\,]), \rho_p(f[\,][\,]) \right\rangle = \| f[\,][\,] \|^2$$

Pattern Matching (Term 2)

$$\boxed{-2\left\langle \rho_p(f[\,][\,]), \rho_p(\chi[\,][\,]) \cdot g[\,][\,] \right\rangle}$$

Using the the fact that $\chi[\,][\,]$ is real-valued, we can move it to the other side of the dot-product:

$$\left\langle \rho_p(f[\,][\,]), \rho_p(\chi[\,][\,]) \cdot g[\,][\,] \right\rangle = \left\langle \rho_p(\chi[\,][\,]) \cdot \rho_p(f[\,][\,]), g[\,][\,] \right\rangle$$

Pattern Matching (Term 2)

$$\boxed{-2\left\langle \rho_p(f[\,][\,]), \rho_p(\chi[\,][\,])\cdot g[\,][\,] \right\rangle}$$

Using the the fact that $\chi[\,][\,]$ is real-valued, we can move it to the other side of the dot-product:

$$\left\langle \rho_p(f[\,][\,]), \rho_p(\chi[\,][\,])\cdot g[\,][\,] \right\rangle = \left\langle \rho_p(\chi[\,][\,])\cdot \rho_p(f[\,][\,]), g[\,][\,] \right\rangle$$

Since the product of the representations is the representation of the products we get:

$$\left\langle \rho_p(\chi[\,][\,])\cdot \rho_p(f[\,][\,]), g[\,][\,] \right\rangle = \left\langle \rho_p(\chi[\,][\,]\cdot f[\,][\,]), g[\,][\,] \right\rangle$$

Pattern Matching (Term 2)

$$\boxed{-2\left\langle \rho_p(f[\,][\,]), \rho_p(\chi[\,][\,])\cdot g[\,][\,]\right\rangle}$$

Using the the fact that $\chi[\,][\,]$ is real-valued, we can move it to the other side of the dot-product:

$$\left\langle \rho_p(f[\,][\,]), \rho_p(\chi[\,][\,])\cdot g[\,][\,]\right\rangle = \left\langle \rho_p(\chi[\,][\,])\cdot \rho_p(f[\,][\,]), g[\,][\,]\right\rangle$$

Since the product of the representations is the representation of the products we get:

$$\left\langle \rho_p(\chi[\,][\,])\cdot \rho_p(f[\,][\,]), g[\,][\,]\right\rangle = \left\langle \rho_p(\chi[\,][\,]\cdot f[\,][\,]), g[\,][\,]\right\rangle$$

And since $\chi[\,][\,]$ is equal to one whenever $f[\,][\,]$ is non-zero we get:

$$\left\langle \rho_p(\chi[\,][\,]\cdot f[\,][\,]), g[\,][\,]\right\rangle = \left\langle \rho_p(f[\,][\,]), g[\,][\,]\right\rangle$$

Pattern Matching (Term 3)

$$\boxed{\left\langle \rho_p(\chi[\,][\,])\cdot g[\,][\,], \rho_p(\chi[\,][\,])\cdot g[\,][\,] \right\rangle}$$

Using the the fact that $\chi[\,][\,]$ and g[ ][ ] are both real-valued, we can move them to the other sides of the dot-product:

$$\left\langle \rho_p(\chi[\,][\,])\cdot g[\,][\,], \rho_p(\chi[\,][\,])\cdot g[\,][\,] \right\rangle = \left\langle \left(\rho_p(\chi[\,][\,])\right)^2, g^2[\,][\,] \right\rangle$$

Pattern Matching (Term 3)

$$\boxed{\left\langle \rho_p(\chi[\,][\,])\cdot g[\,][\,],\rho_p(\chi[\,][\,])\cdot g[\,][\,]\right\rangle}$$

Using the the fact that $\chi[\,][\,]$ and g[ ][ ] are both real-valued, we can move them to the other side of the dot-product:

$$\left\langle \rho_p(\chi[\,][\,])\cdot g[\,][\,],\rho_p(\chi[\,][\,])\cdot g[\,][\,]\right\rangle=\left\langle \left(\rho_p(\chi[\,][\,])\right)^2,g^2[\,][\,]\right\rangle$$

Since the $\chi[\,][\,]$ is always equal to either 0 or 1, we have $\chi[\,][\,]=\chi^2[\,][\,]$ so that:

$$\left\langle \left(\rho_p(\chi[\,][\,])\right)^2,g^2[\,][\,]\right\rangle=\left\langle \rho_p(\chi[\,][\,]),g^2[\,][\,]\right\rangle$$

Pattern Matching

Combining all of this together, we get:

$$\begin{aligned}\left\|\rho_p(f[\ ][\ ]) - \rho_p(\chi[\ ][\ ])\cdot g[\ ][\ ]\right\|^2 = \|f[\ ][\ ]\|^2 &+ \left\langle \rho_p(\chi[\ ][\ ]), g^2[\ ][\ ]\right\rangle \\ &- 2\left\langle \rho_p(f[\ ][\ ]), g[\ ][\ ]\right\rangle\end{aligned}$$

Pattern Matching

Combining all of this together, we get:

$$\left\|\rho_p(f[\,][\,]) - \rho_p(\chi[\,][\,])\cdot g[\,][\,]\right\|^2 = \left\|f[\,][\,]\right\|^2 + \left\langle \rho_p(\chi[\,][\,]), g^2[\,][\,]\right\rangle - 2\left\langle \rho_p(f[\,][\,]), g[\,][\,]\right\rangle$$

Or somewhat more cleanly:

$$\left\|f[\,][\,]\right\|^2 + \chi[\,][\,] * g^2[\,][\,] - 2f[\,][\,] * g[\,][\,]$$

Pattern Matching

Combining all of this together, we get:

$$\left\| \rho_p(f[\,][\,]) - \rho_p(\chi[\,][\,]) \cdot g[\,][\,] \right\|^2 = \| f[\,][\,] \|^2 + \left\langle \rho_p(\chi[\,][\,]), g^2[\,][\,] \right\rangle \\ - 2\left\langle \rho_p(f[\,][\,]), g[\,][\,] \right\rangle$$

Or somewhat more cleanly:

$$\| f[\,][\,] \|^2 + \underbrace{\boxed{\chi[\,][\,] * g^2[\,][\,]}}_{\boxed{\text{The windowed norm}}} - \underbrace{\boxed{2 f[\,][\,] * g[\,][\,]}}_{\boxed{\text{The moving dot-product}}}$$

Pattern Matching

$$\| f[\,][\,] \|^2 + \chi[\,][\,] * g^2[\,][\,] - 2 f[\,][\,] * g[\,][\,]$$