CN550: Neural and Computational Models of Recognition, Memory, and Attention

Invariant Pattern Recognition Methods

Lecture 13

Instructors: Anatoli Gorchetchnikov <anatoli@cns.bu.edu>
Heather Ames <starfly@cns.bu.edu>

Formal Definition of Pattern Recognition

Let's say we have a set of input features $\boldsymbol{X}$

Any pattern f will be some function of the $\boldsymbol{X}$: $f(X)$

A set of all possible patterns f is denoted as V

A general problem of pattern recognition is to construct a function $s(f)$ such that for all patterns in V $s(f)=c(f)$ where $c(f)$ is a desired classification

This is a simplification, the same pattern can have different desired classification depending on the context

Basics of Group Theory

Group is a set G with an operation on this set * that satisfies the following criteria:

Closure: if a and b are in G then $c=a*b$ is also in G

Associativity: for all a, b, and c in G $(a*b)*c=a*(b*c)$

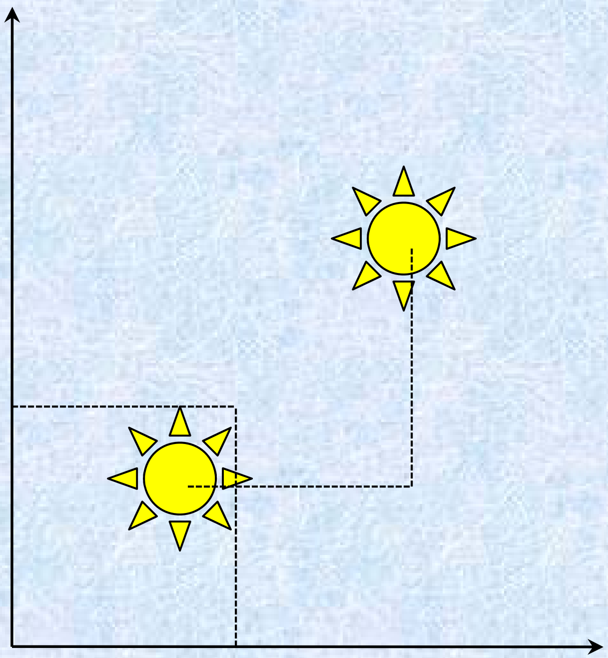
Identity element: there exist e in G such that for any a in G $a*e=e*a=a$

Inverse element: for any a in G there exist b in G such that $a*b=b*a=e$

Note that in general commutativity $a*b=b*a$ is not required

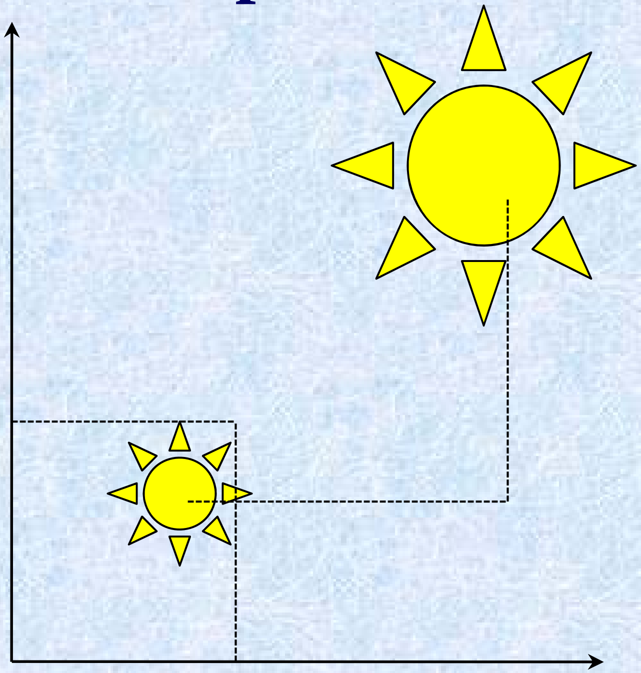
Translation in Terms of Groups

- Elements – cartesian coordinates
- Operation – addition
- Closure: satisfied, adding two coordinates results in another coordinate
- Associativity: satisfied, follows from associativity of addition
- Identity element: origin
- Inverse element: symmetric point



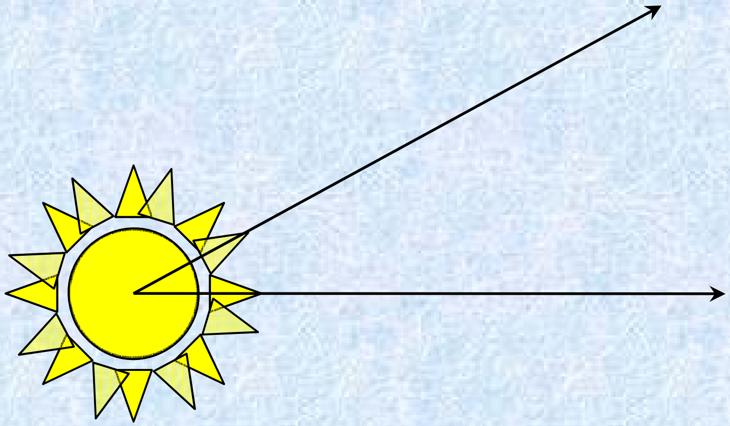
Scale in Terms of Groups

- Elements – cartesian coordinates
- Operation – multiplication
- Closure: satisfied, multiplying two coordinates results in another coordinate
- Associativity: satisfied, follows from associativity of multiplication
- Identity element: point (1,1,…)
- Inverse element: point with reciprocal coordinates



- Works better if the pattern is centered on origin
- Note that using log space for coordinates makes it identical to translation group

Rotation in Terms of Groups



Elements – polar coordinates

Operation – addition on theta

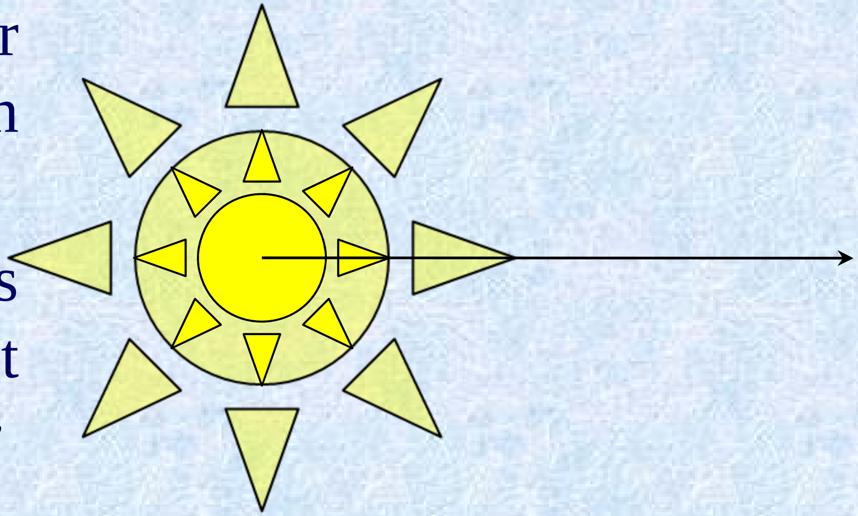
Closure: satisfied

Associativity: satisfied

Identity element: $\theta=0$

Inverse element: $\theta' = -\theta$

Only works well if the center of rotation is used as an origin



Uniform scale also works well in this setting, except we use multiplication on r

Preliminary Thoughts

Both rotation and scaling invariance are easier to achieve if we work with coordinate system centered on the object

Translation and rotation involve addition, scaling involves multiplication, but can be reduced to addition by using log coordinates

Uniform scaling works well in both polar and cartesian coordinates, non-uniform scaling is easier in cartesian coordinates

Having discrete space (pixels in digital images) complicates the problem by violating the closure requirement

Invariant Pattern Recognition

Let's say we have a set of input features $\boldsymbol{X}$

Any pattern f will be some function of the $\boldsymbol{X}$: $f(\boldsymbol{X})$

A set of all possible patterns f is denoted as $\boldsymbol{V}$

Let G be a group that acts on set $\boldsymbol{X}$

The desired classification c is invariant with respect to the action of the group: $c(gf)=c(f)$

So now we need to construct a classifier s such that

$$s(gf)=c(f)$$

Most common invariances that we are interested in are: rotation, translation, and scale

Two General Approaches

First convert features to their invariant counterparts, then do the classification: instead of $s(gf(\mathbf{X}))$ do $s(f(g'\mathbf{X}))$

Advantage: can use any conventional and simple pattern classifier

Combine both stages, essentially creating a parametrized invariant and classify based on these parameters

Advantage: fits naturally in neural network paradigm

Examples of this approach are Neocognitron and ARTSCAN

Invariant Feature Extraction

Two main classes:

- Based on integral transforms

$$z(w) = \int_X f(x)k(w,x)dx$$

  - Converts features x into invariant features *z(w)*
  - If ***X*** is finite use sum in place of integral
  - Must be invertible
  - The modulus of the transform *z(w)* must be invariant under the action of group

- Based on moments – weighted sums of the pattern f over the entire input field

Fourier Transform

Advantages:

It has been proven that any integral transform used to produce invariance for certain classes of groups (Lie groups) has a form of Fourier transform

Note that Lie groups are smooth, differentiable, and cover a lot of geometrical spatial properties, e.g. a unit circle in a complex plane is a Lie group

Using Fast Fourier Transform algorithm (FFT) gives a computational edge over other methods ($O(n \log n)$)

Fourier Transform

$$z(w) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i w x} dx$$

Fourier transform is an invertible integral transform

It is also invariant under the translation of *f(x)*

Circular Fourier transform

$$z(r) = \frac{1}{2\pi} \int_{0}^{2\pi} f(r,\theta) e^{-il\theta} d\theta$$

is invariant under rotation (translation of θ)

Note that we are back to our initial insight that rotation is a translation in polar coordinates

Mellin Transform

If *f(r)* is a function of a positive real line and *u* is a complex variable

$$z(u) = \int_0^\infty f(r) r^{u-1} dr$$

For imaginary *u*

$$z(w) = \int_0^\infty f(r) r^{iw-1} dr$$

Using a variable substitution $r=e^{-2\pi x}$ it is turned into a Fourier transform: so Mellin transform is a Fourier transform evaluated over the exponential scale

Remember that scale invariance is the same as translation invariance in log space

Fourier-Mellin Transform

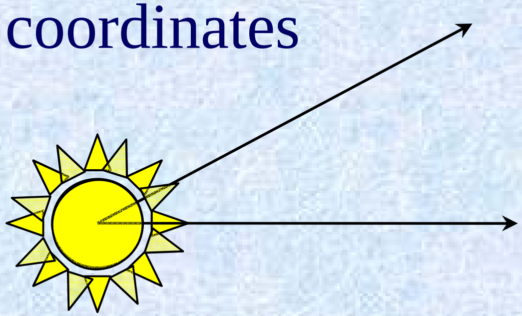
Combining circular Fourier and radial Mellin transforms

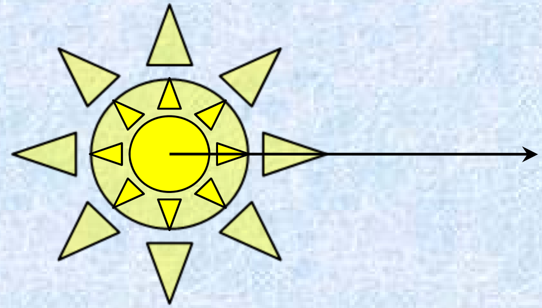
$$z(l, w) = \frac{1}{2\pi} \int_0^{\infty} \int_0^{2\pi} f(r, \theta) e^{-il\theta} r^{iw-1} d\theta dr$$

By using log-polar mapping we convert it into 2D Fourier transform and thus get z invariant to both rotation and scaling

This system performs well under noise

Technically it is nothing more than an implementation of our original discussion of invariance for a centered object in polar coordinates





Procedure for RTS Invariance

Use 2D Fourier transform; its power spectrum is invariant under translation

Convert it to polar coordinates and perform log-polar mapping

Calculate another 2D Fourier transform; its result will additionally be rotation and scale invariant

Aside: retina does log-polar mapping for us but not in-between stages

There are other transforms out there that achieve functions similar to these we have discussed

Algebraic Invariants

A moment is a weighted sum of a pattern over the entire input field

$$m_{p,q} = \iint_{-\infty} x^p y^q f(x,y)\,dxdy$$

Here p and q are non-negative integers, $p+q$ is the order of moment m and pattern $f(x,y)$ is piecewise continuous function that is non-zero only on a finite part of the plane

Moments of different order characterize the distribution of the pattern $f(x,y)$ in space

Thus if we can choose the appropriate moments we can extract the features of this distribution that are rotation, translation and scale invariant

Algebraic Invariants

$$m_{p,q} = \iint_{-\infty} x^p y^q f(x,y)\,dxdy$$

The centroid of f is defined by using zero and first order moments as

$$x_0 = \frac{m_{1,0}}{m_{0,0}}; y_0 = \frac{m_{0,1}}{m_{0,0}}$$

If we translate f by some vector, then centroid is also translated, so it is translation invariant with respect to f

So we can construct moments with respect to this centroid

Central Moments

We can define central moments

$$v_{p,q} = \iint_{-\infty} x^p y^q f(x + x_0, y + y_0) dxdy$$

as the moments of a centralized pattern, and by definition they are translation invariant

Another way to write them

$$v_{p,q} = \iint_{-\infty} (x - x_0)^p (y - y_0)^q f(x, y) dxdy$$

Normalized and Rotation Invariant Moments

Now we can define normalized moments

$$\mu_{p,q} = \frac{v_{p,q}}{v_{0,0}^{1+(p+q)/2}}$$

These are additionally scale invariant

Finally, we can also derive moments that are rotation invariant, e.g.

$$\phi_1 = \mu_{2,0} + \mu_{0,2}$$

$$\phi_2 = (\mu_{2,0} - \mu_{0,2})^2 + 4\mu_{1,1}^2$$

These are two of the set of seven that is called Hu's moments

General theory has shown that this set is not complete

Other Moments

Regular moments are highly noise sensitive

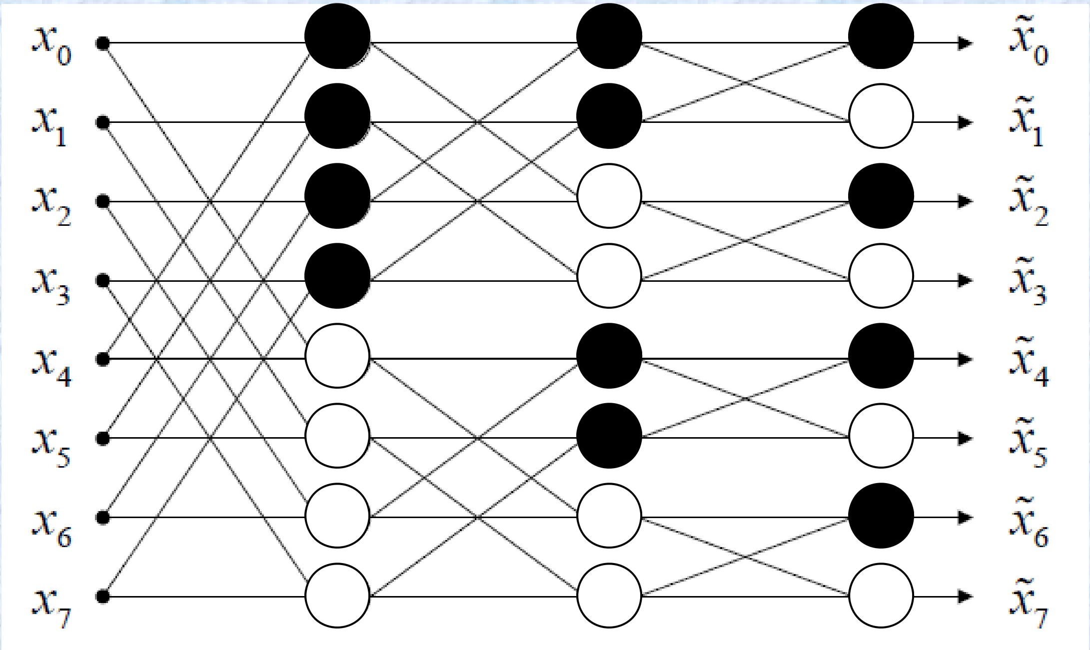
There are many sets of different moments that has been used for invariant pattern recognition:

- Legendre
- Rotational
- Complex
- Zernike

Out of these Zernike moments seem to be the best

Downside: moments are not always easy to compute

Fast Translation Invariant Transform



- Black nodes use f_1
- White nodes use f_2
- f_1 and f_2 can be any commutative functions of two inputs
- The output is invariant under cyclic translation
- R-transform is defined as $f_1=x_1+x_2 \quad f_2=|x_1-x_2|$
- M-transform is defined as $f_1=max(x_1,x_2)$ $f_2=min(x_1,x_2)$
- Correlation transform has $f_1=x_1+x_2$ $f_2=x_1x_2$
- Do not discriminate fully (e.g. bitwise addition is not detected)

System Requirements

Invariance vs tolerance:

- perfect invariance is hard to achieve due to some computational issues (e.g. round-off errors in continuous functions)
- transformation tolerance is more practical, often satisfactory, and in many cases closer to biology (read upside down text)

Discriminability:

- system has to distinguish patterns that are not related by transformation as well as bring together patterns that are
- invariants of two patterns f_1 and f_2 must be identical if and only if there is a transformation $f_1 = g f_2$

System Requirements

Computational complexity:

- system shall take reasonable amount of time and space to compute s

Ease of training:

- system will be trained on a finite sample
- preferably the sample size should not be too large and the training process too complex

For a biological model both of these requirements are merged together and a desire is added to preserve stability of recognition during on-line learning

System Requirements

Generalization ability:

- related to ease of training
- shall be noise tolerant and generalize to previously unseen patterns

Flexibility:

- ideally shall work for different transformation groups and for different problems
- trade-off with simplicity: it is hard to build simple and flexible system