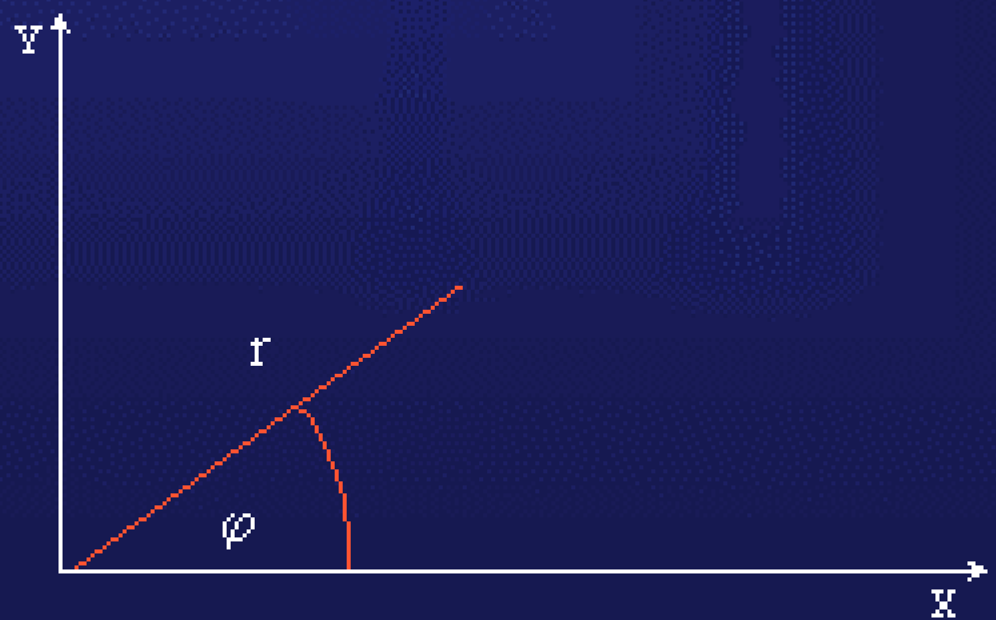
Hough Transform

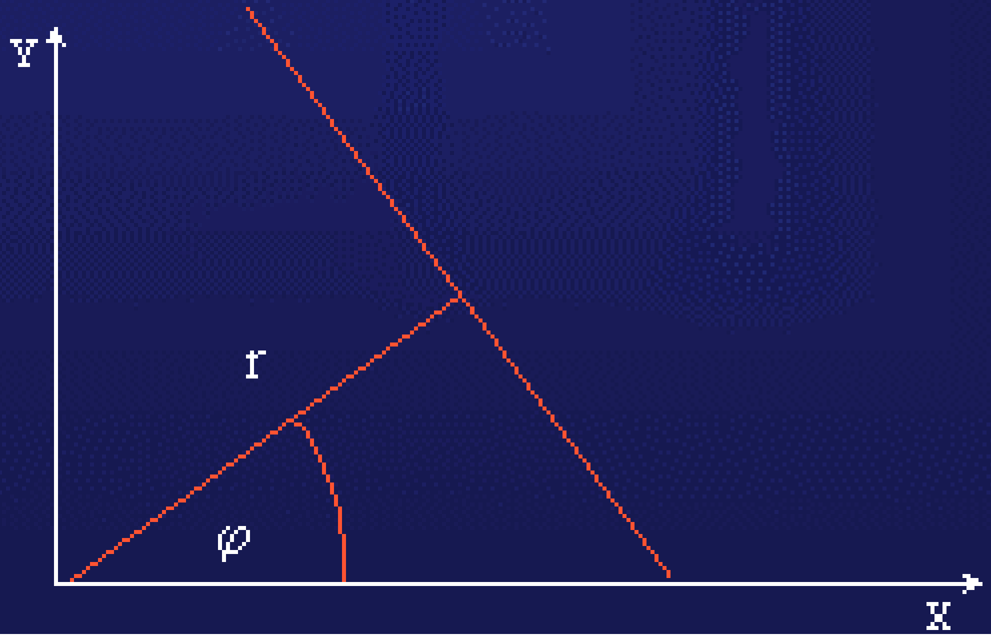
Let us go deeper to understand the principles of creating any kind of Hough Transforms…..

SHT: How it Works

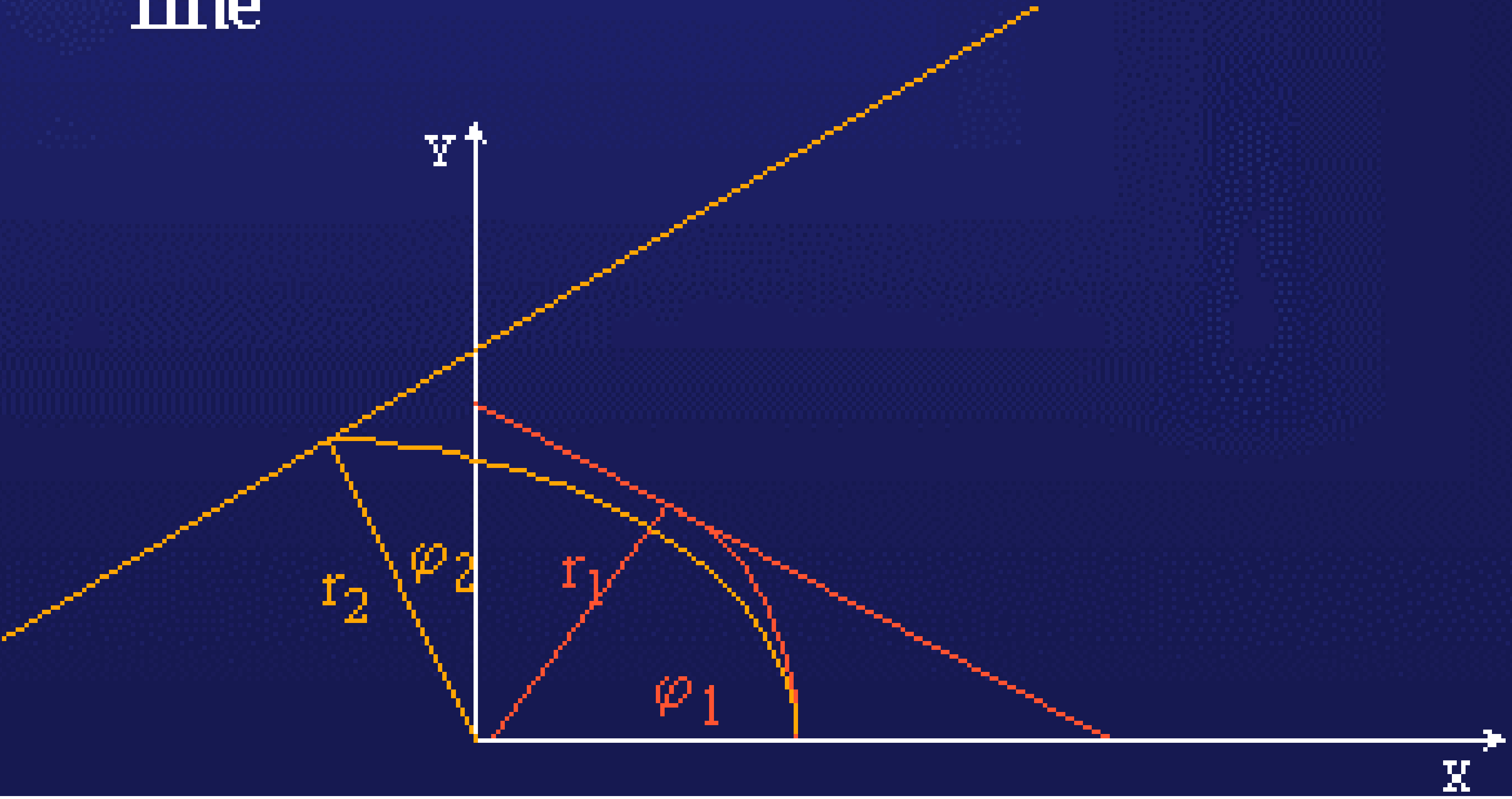
- Usually we are given (r,φ)



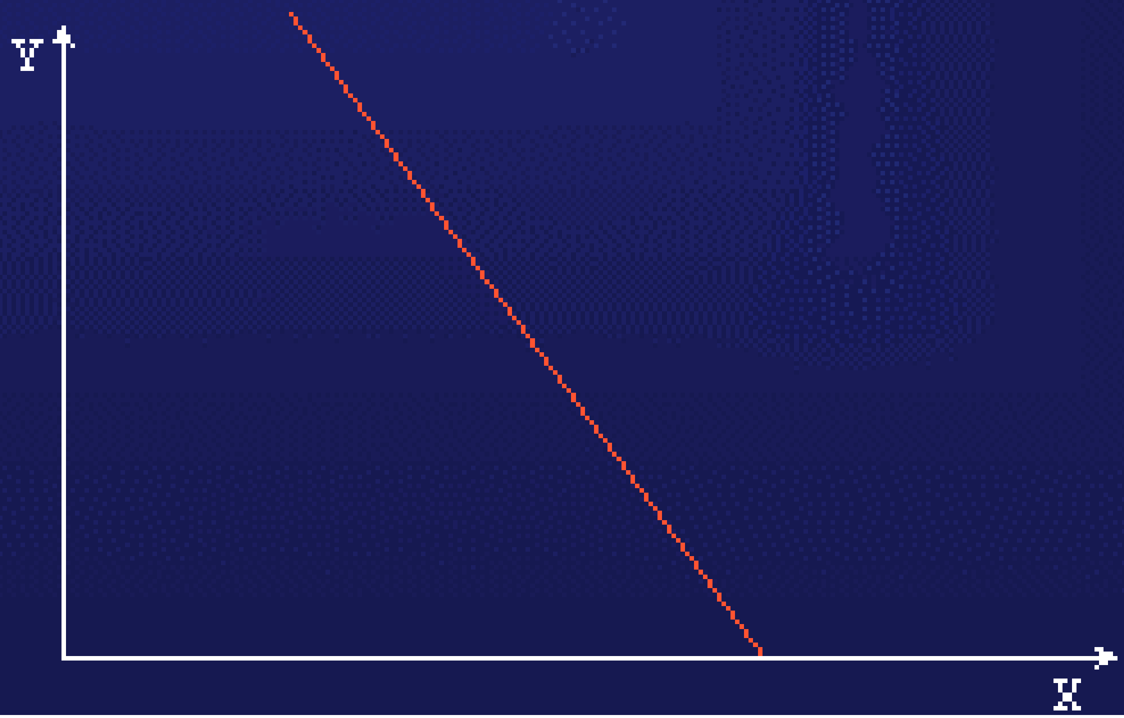
- Usually we are given (r,φ) and must calculate all (x,y) points



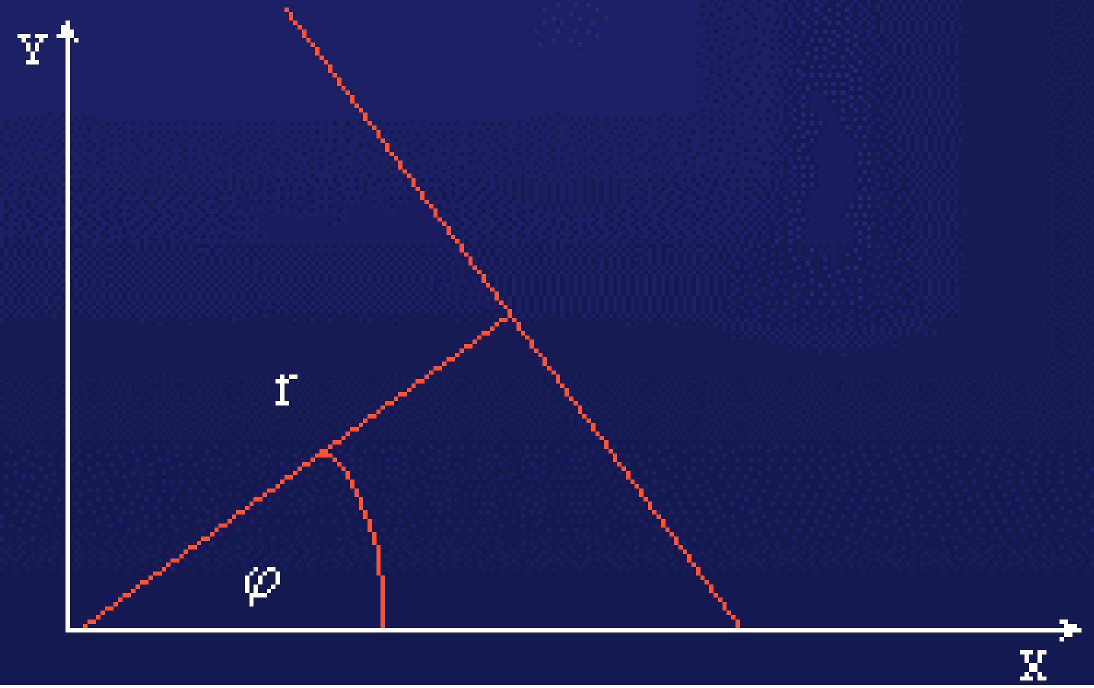
- Therefore given different (r,φ) we get different lines i.e. (x,y) points i.e. (r,φ) parametrise the line



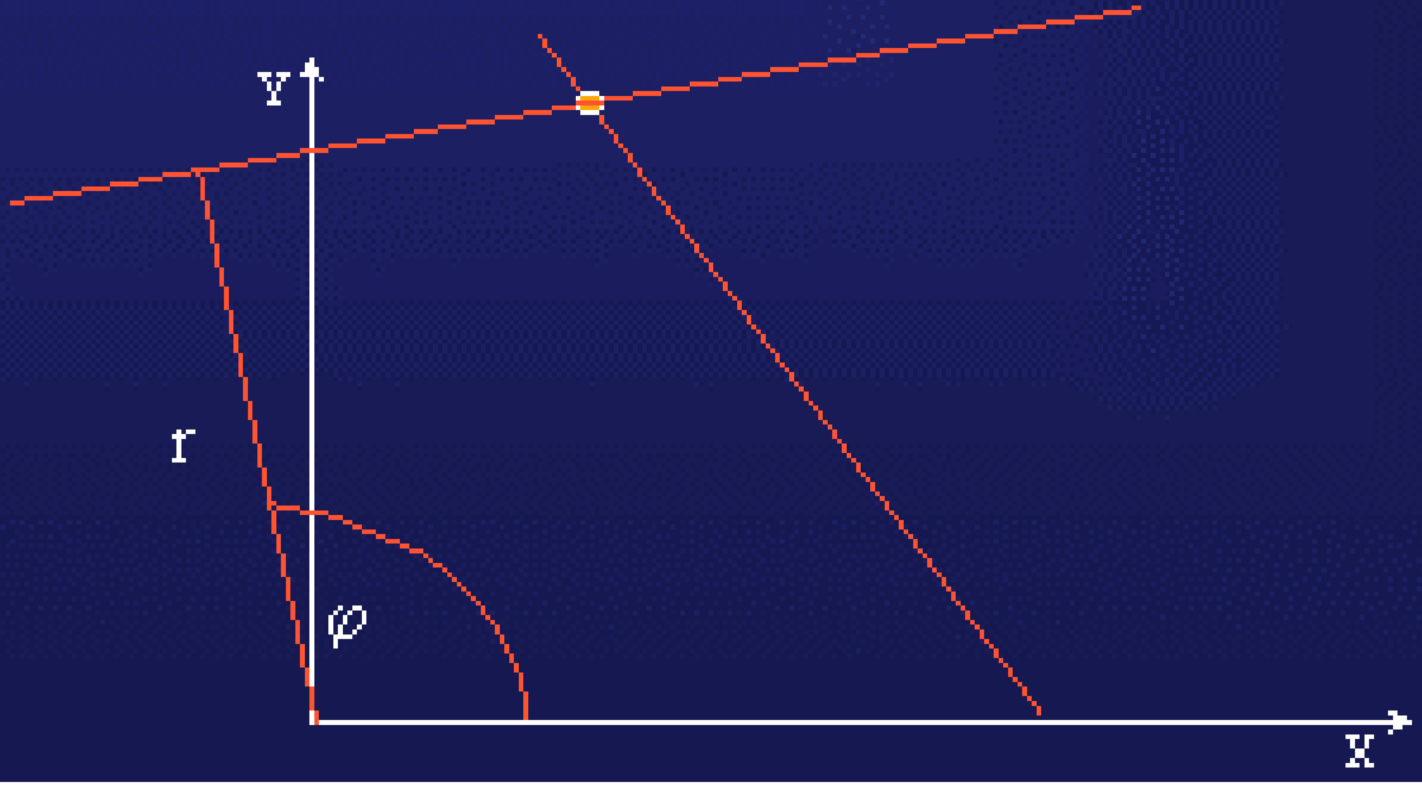
- But from an image analysis point of view we already have (x,y) points



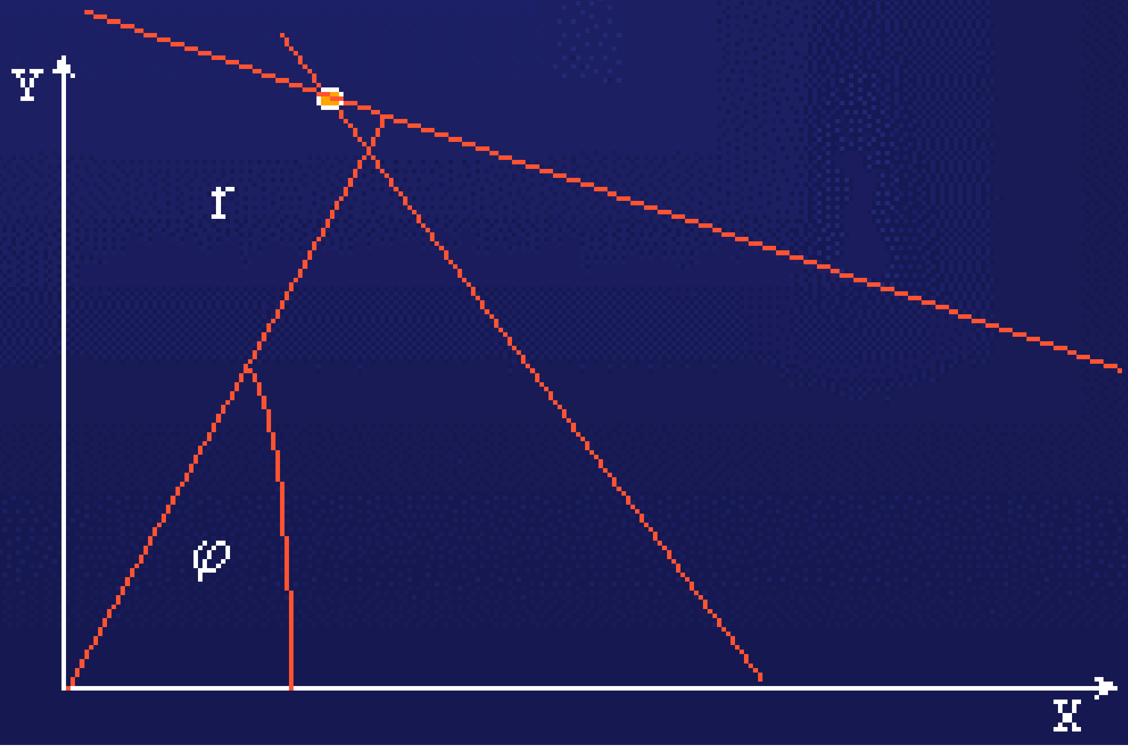
- But from an image analysis point of view we already have (x,y) points and we want (r,φ)



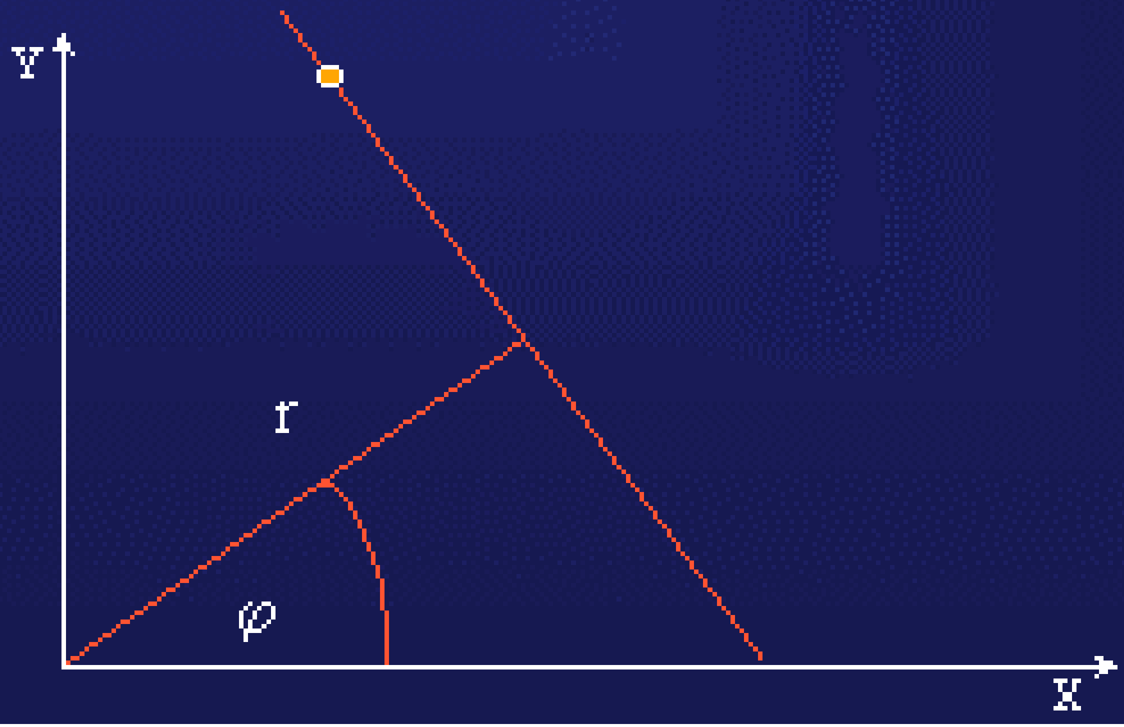
- For each (x,y) point we generate all possible (r,φ) i.e. r and φ are now the variables



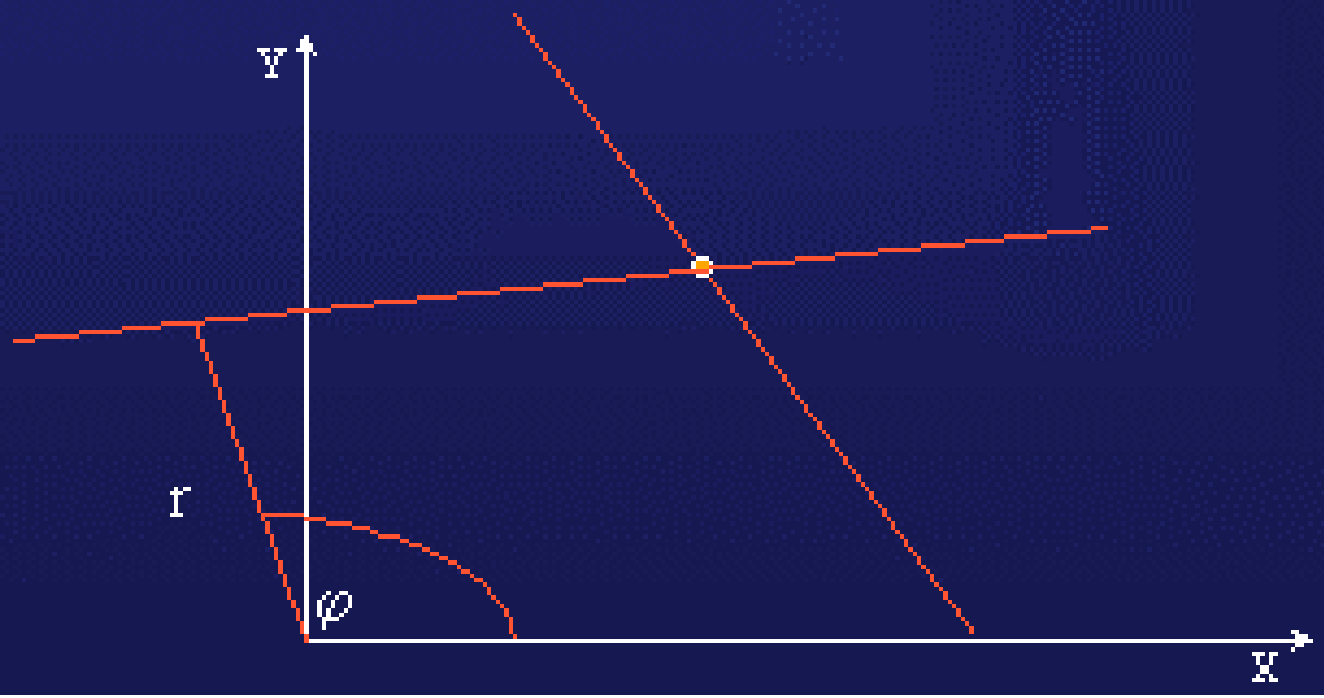
- That is to say that for each point we ask, what are all the possible line passing through this point



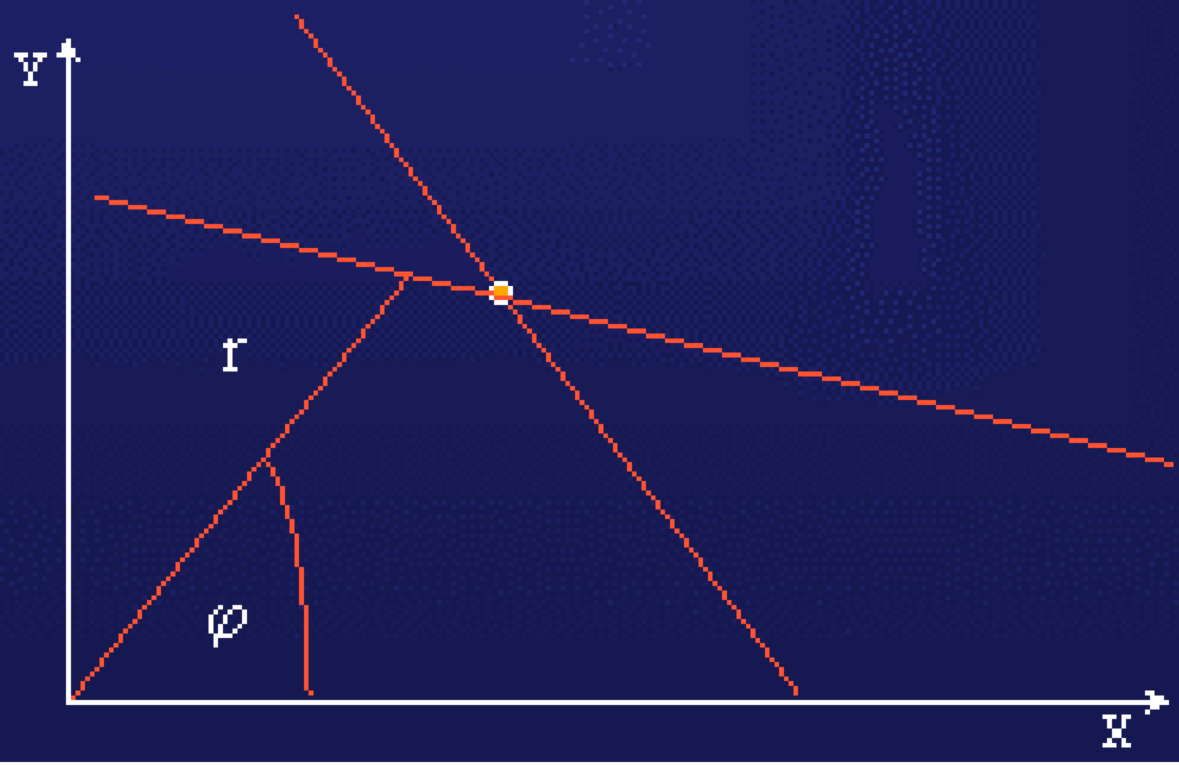
- For each possible line we calculate, we take note of it (i.e its (r,φ) point).



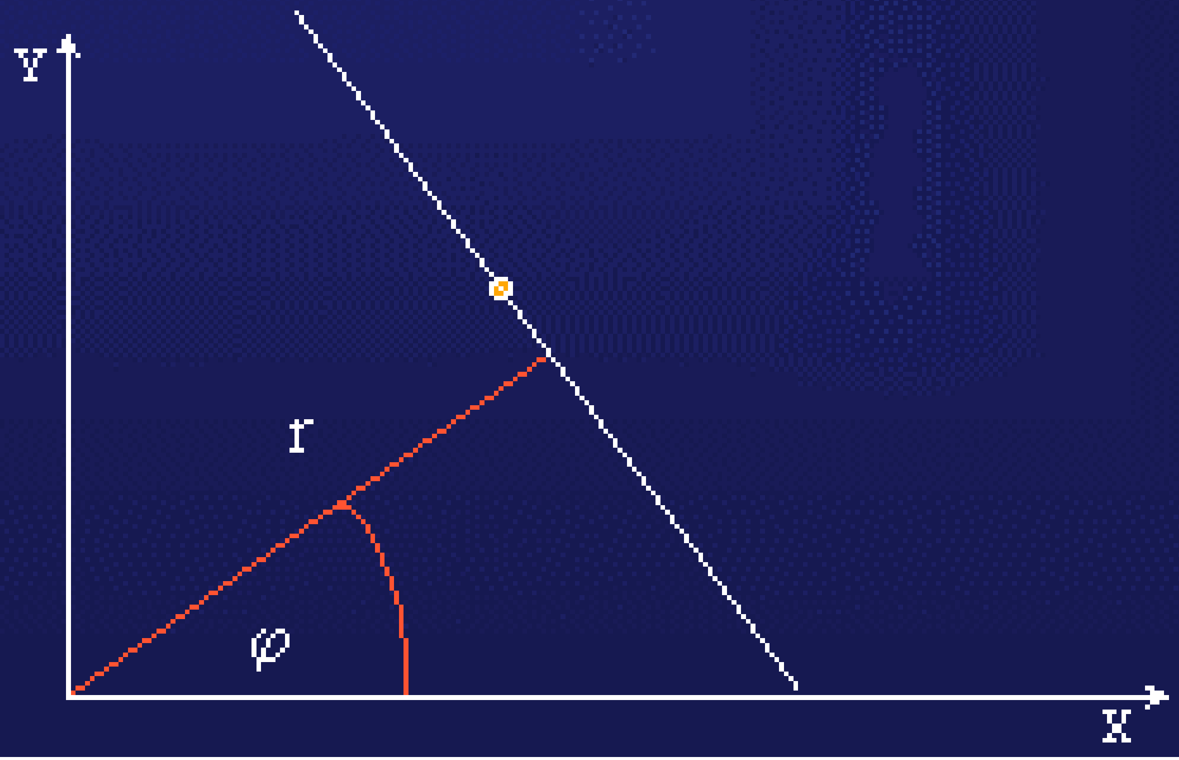
- At the end of this process we have a list of (r,φ) points for each (x,y) point



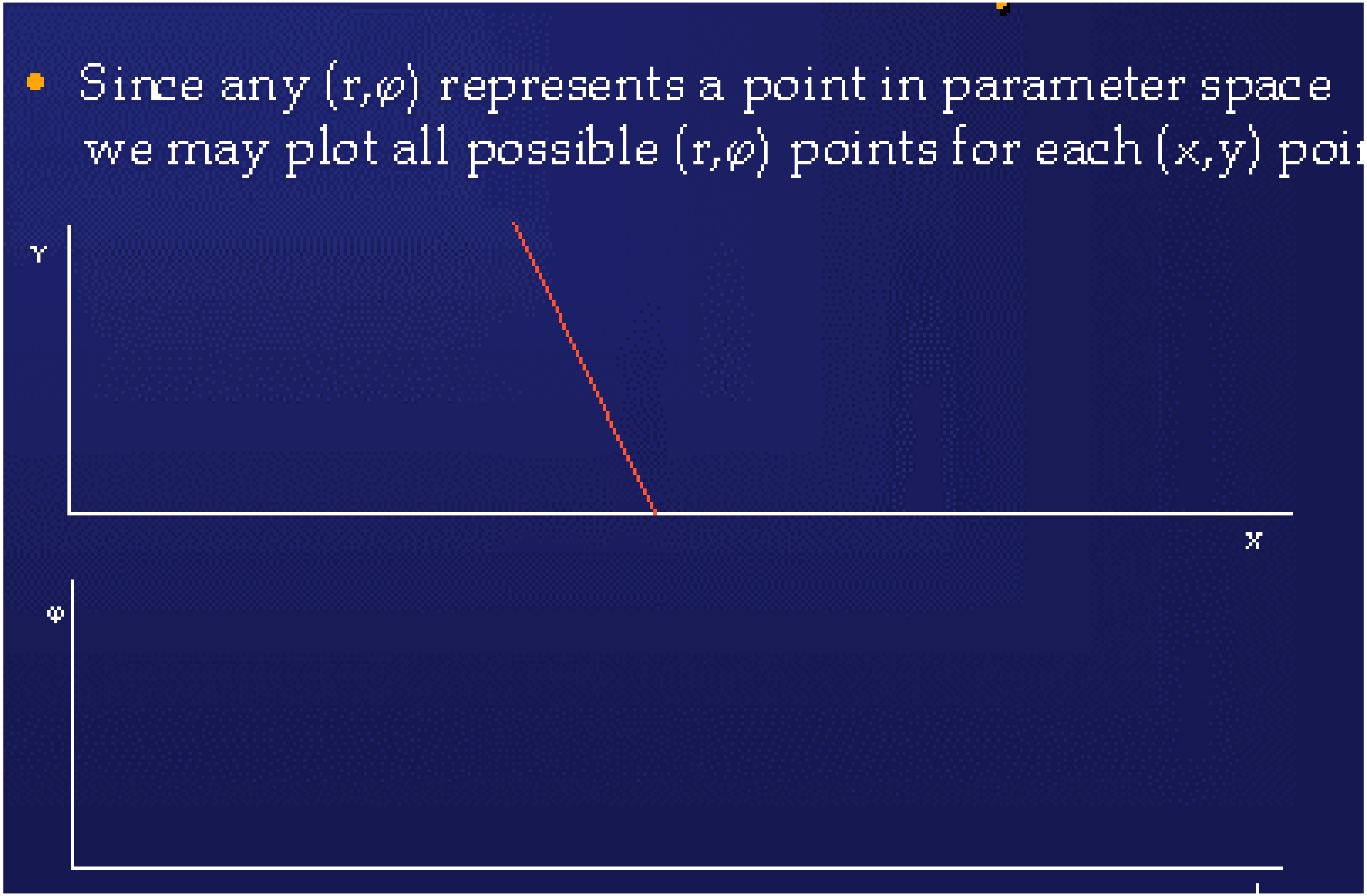
- If a set of (x,y) points are co-linear there will be specific (r,φ) point common to them all



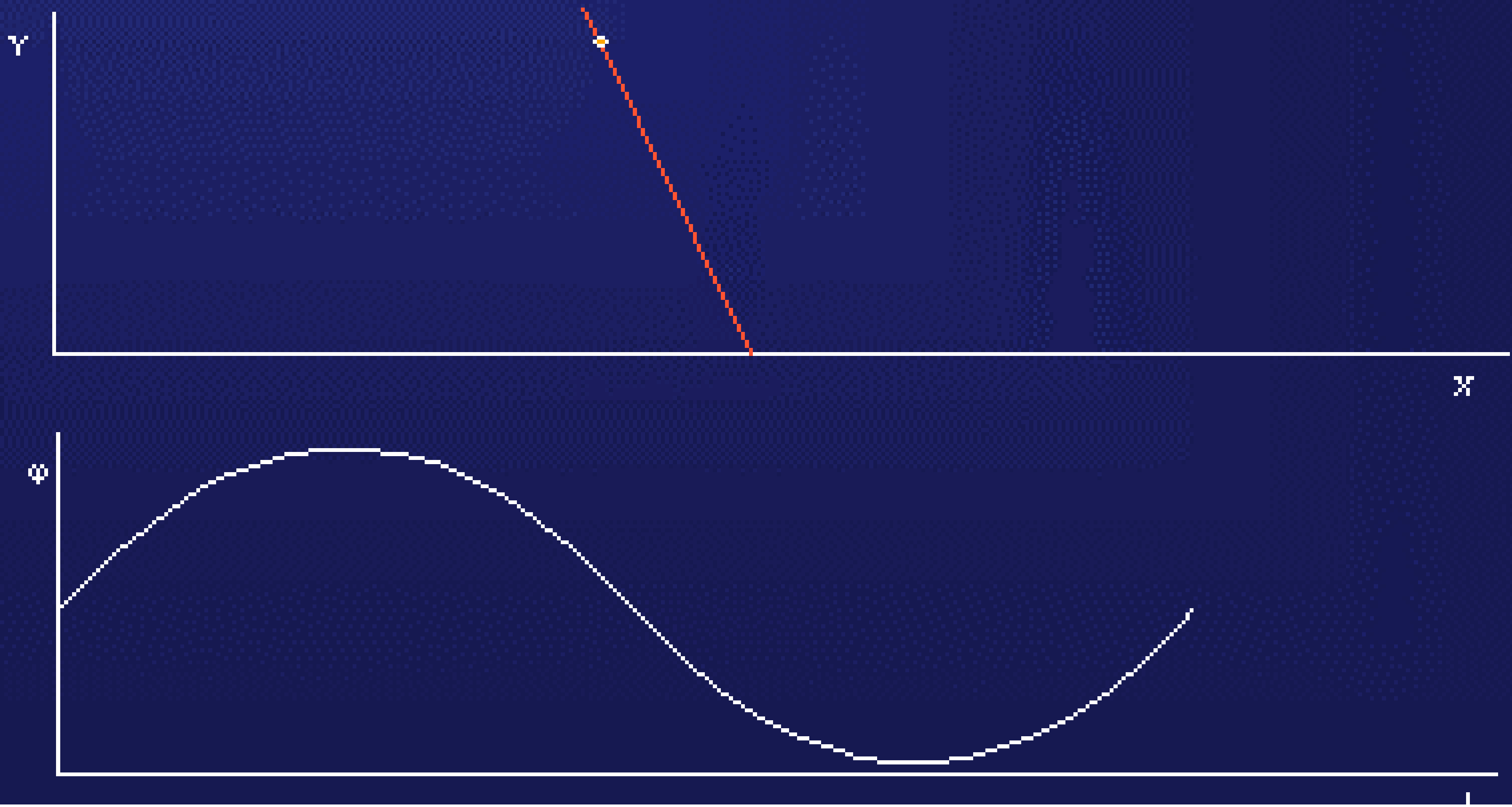
- Isolating these common (r,φ) points then gives us the lines in the image



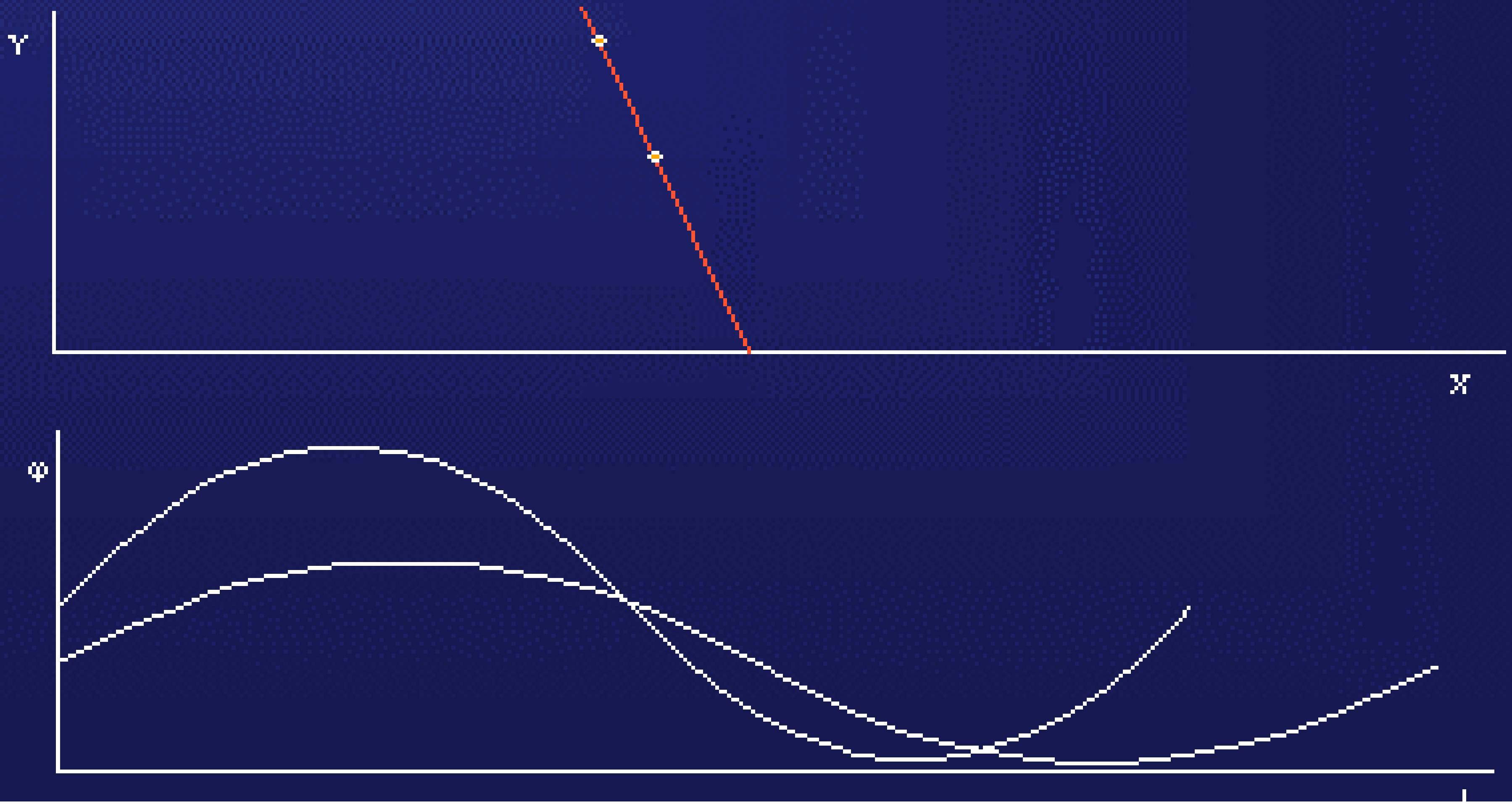
SHT: Another Viewpoint



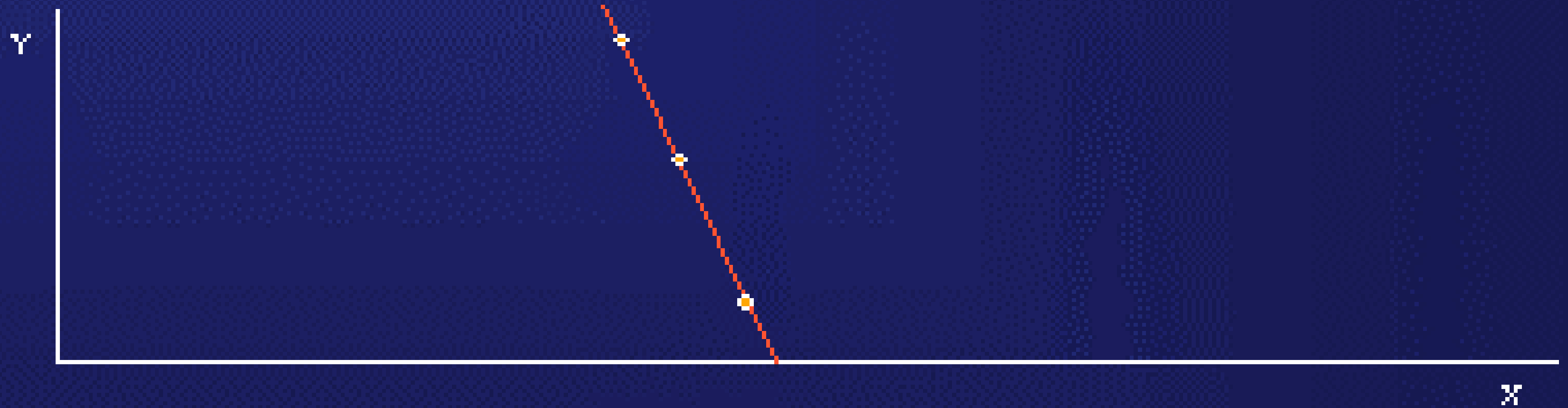
- Taking any particular (x,y) point its set of (r,φ) points is a sinusoid through parameter space

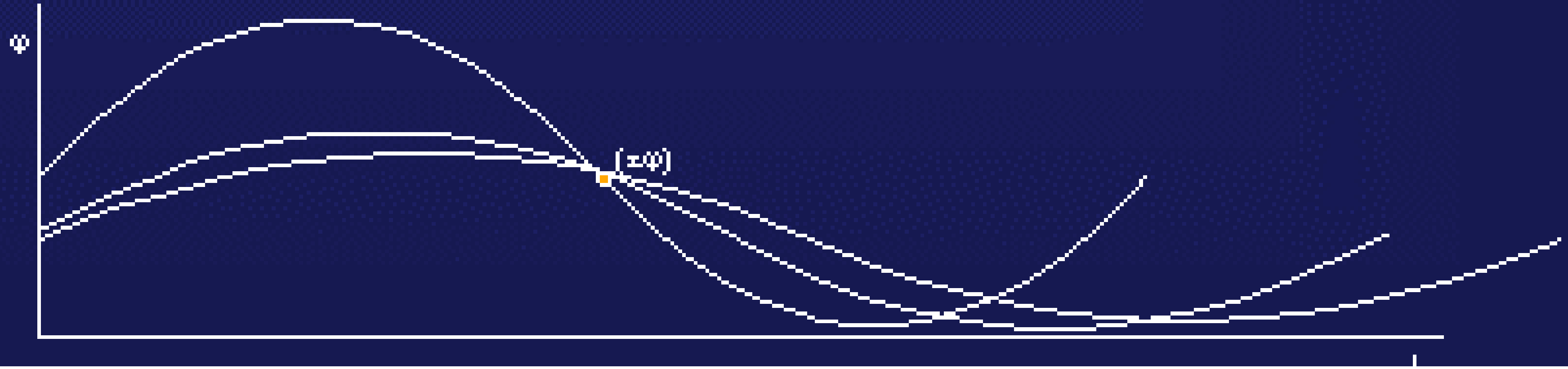


- Each sinusoid tells us every possible line that can pass through the corresponding (x,y) point

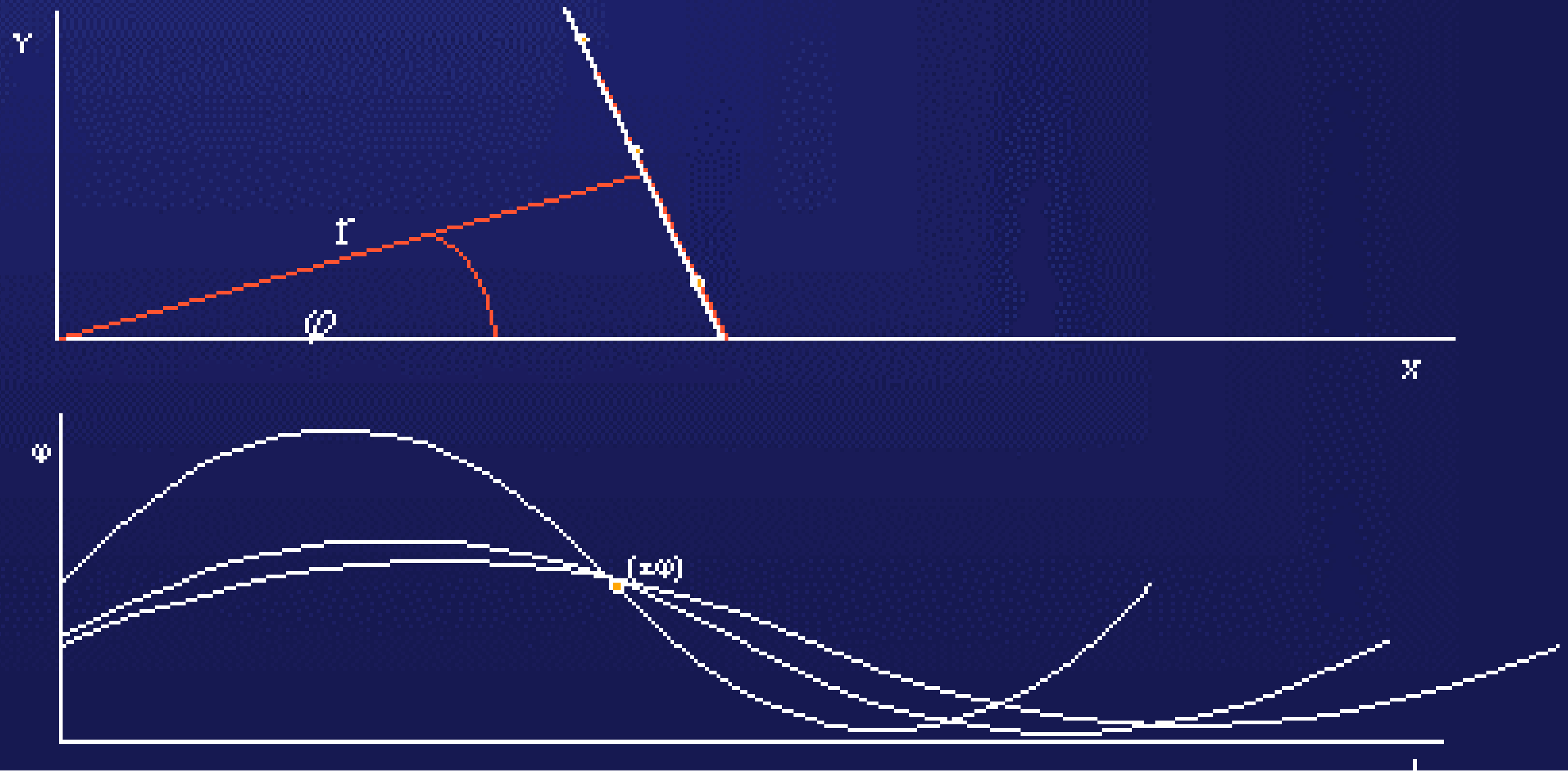


- The line upon which the (x,y) points lie is then given by the (r,φ) pair on which all the sinusoids *agree* (i.e. intersect)



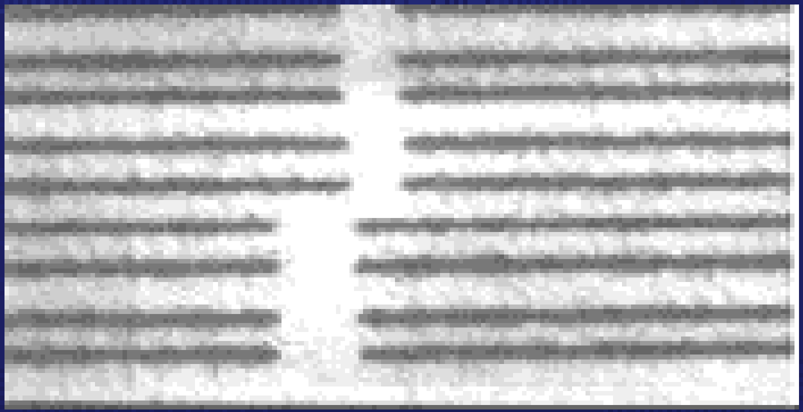


- Finding this intersection we then have the required (r,φ) parameters and therefore the line in the image

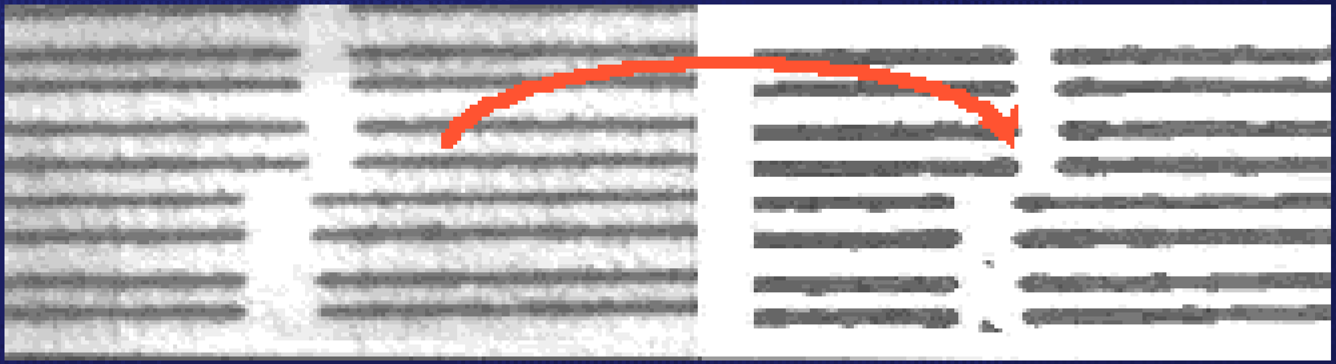


SHT : Example

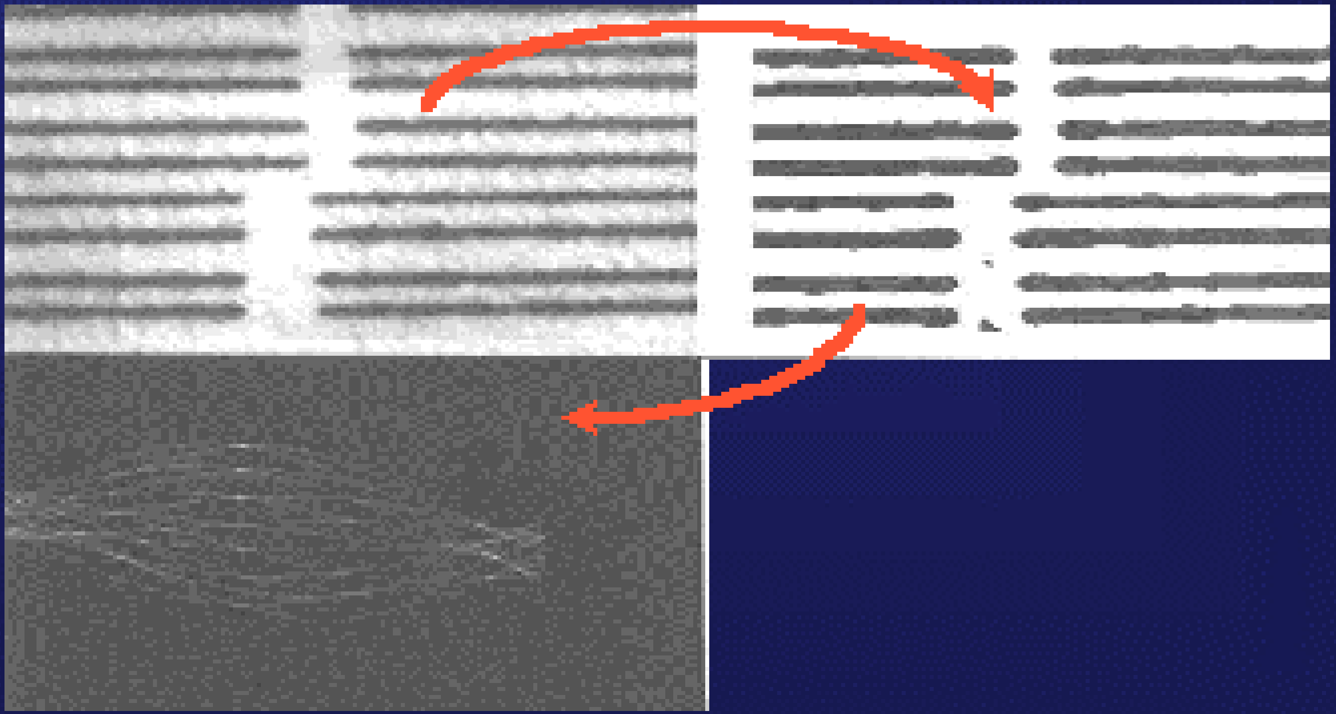
- Given a greyscale image we wish to find all occurences of lines in it



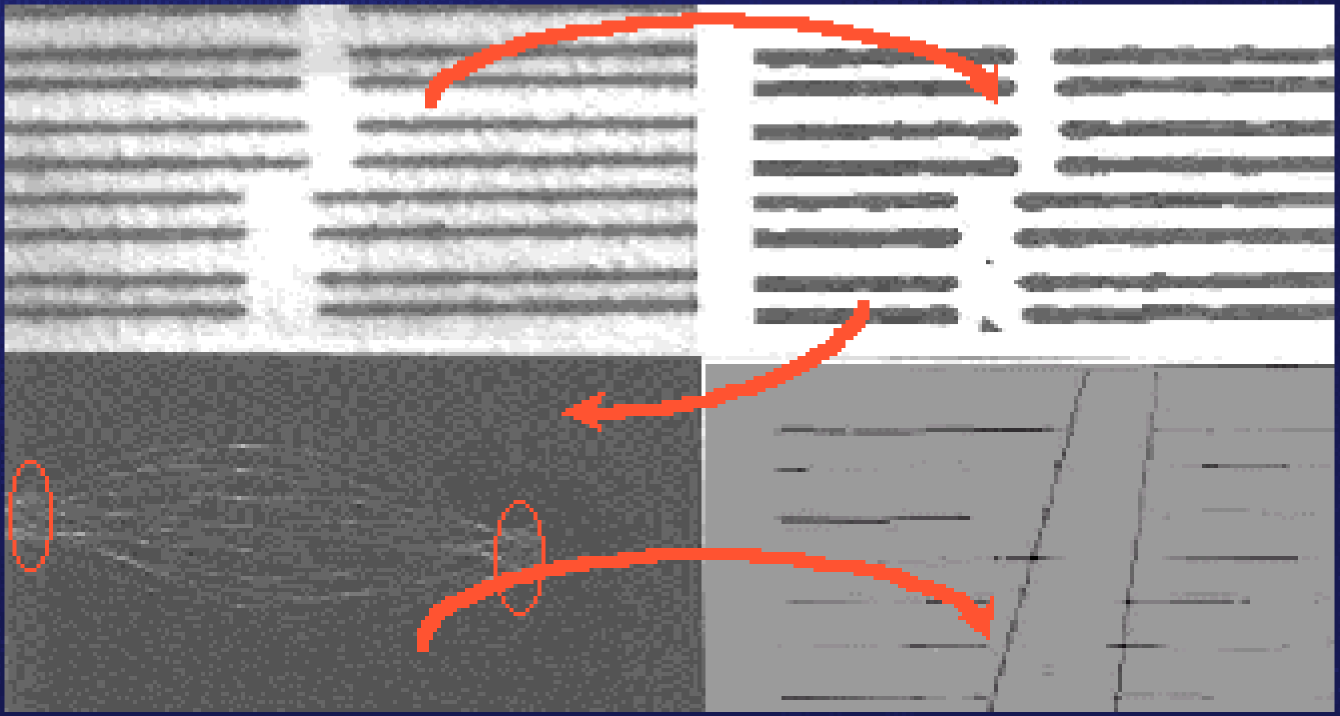
- First we threshold the image and to ease the extraction of all the edges points



- Next we calculate each edge point's corresponding sinusoid in Parameter/Hough Space



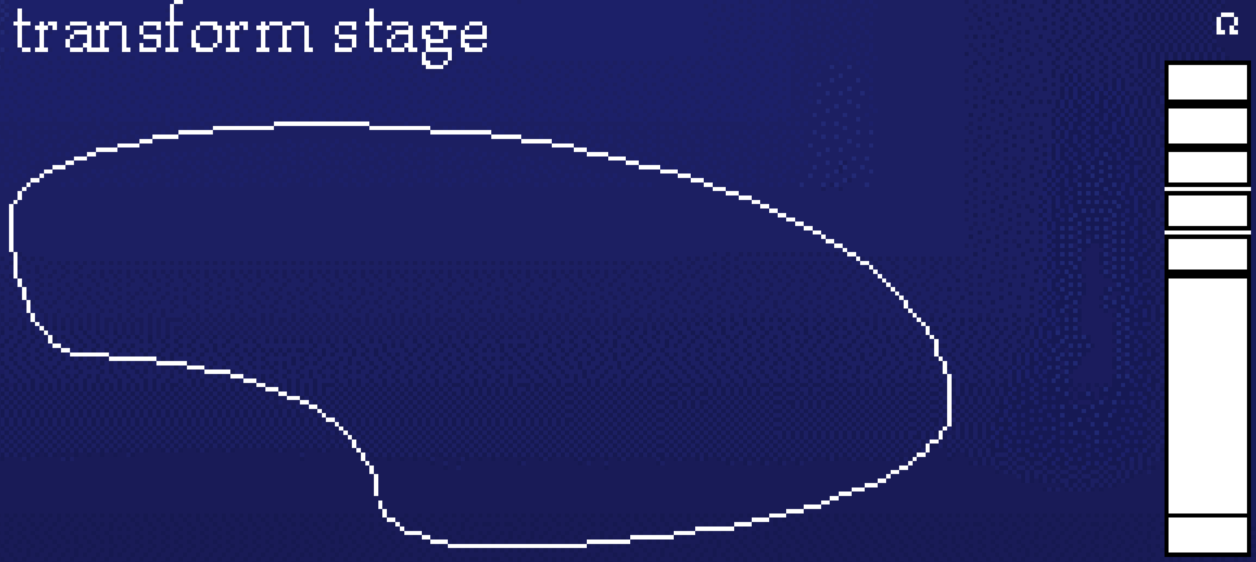
- Isolation of the intersections of the sinusoids, based on a given threshold, then yields the sought (r,φ) pairs



Generalized Hough Transform

SNHT: Building the R-Table

- Given a shape we wish to localise the first step is to build a look-up table, known as a R-table, which will replace the need for a parametric equation in the transform stage

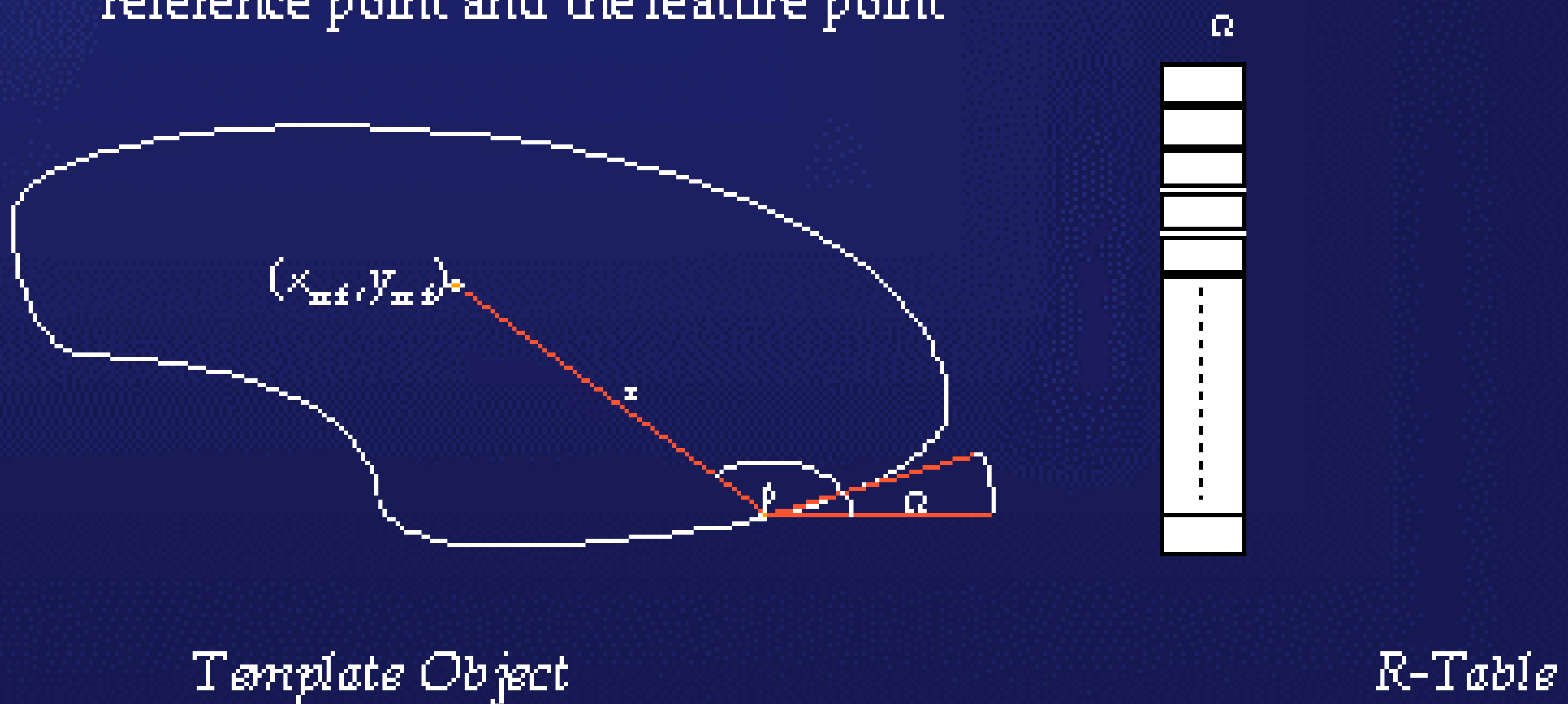


Template Object

R-Table

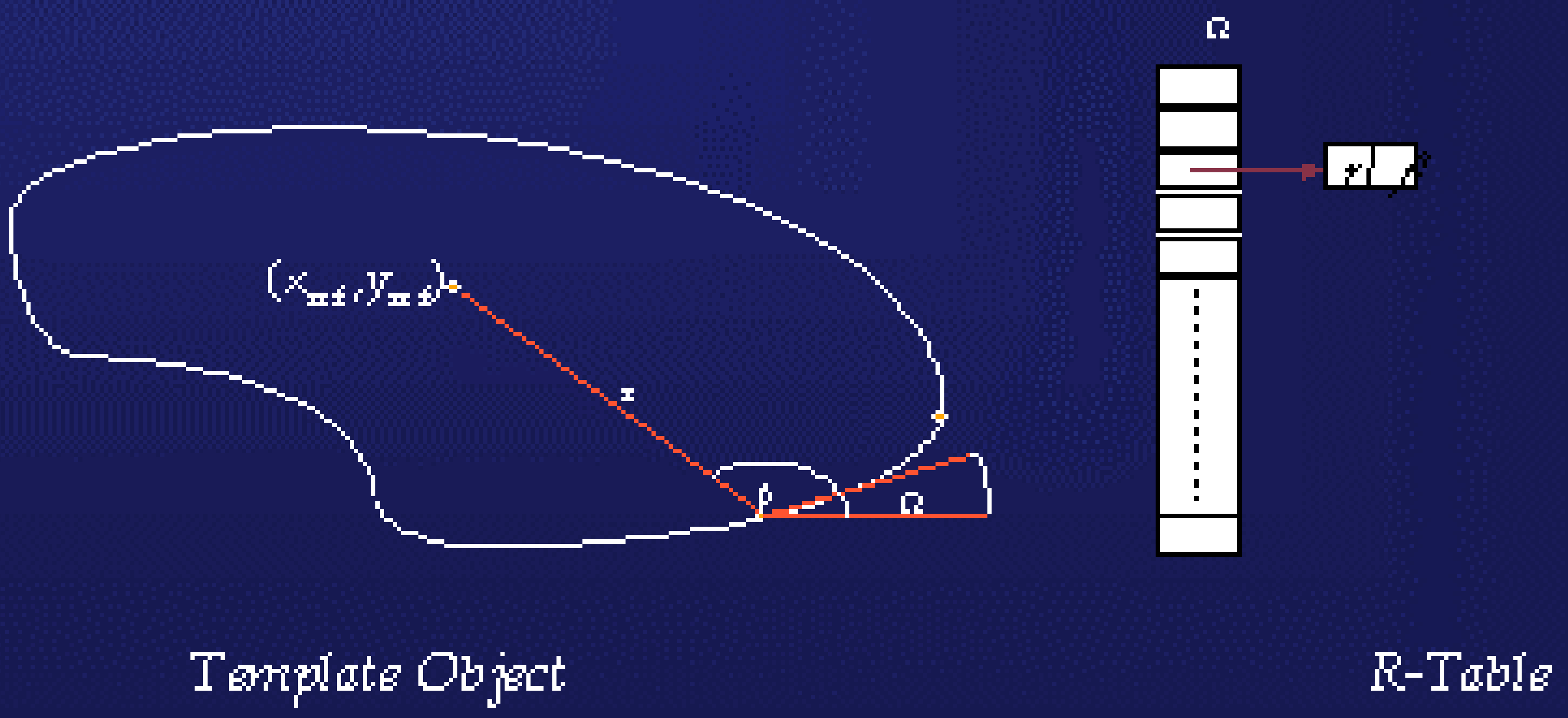
GHT: Building the R-Table

- For each object *feature point* we calculate
  - the orientation, Ω, of the tangential line at that point and
  - the length, r, and orientation, β, of the radial vector joining the reference point and the feature point



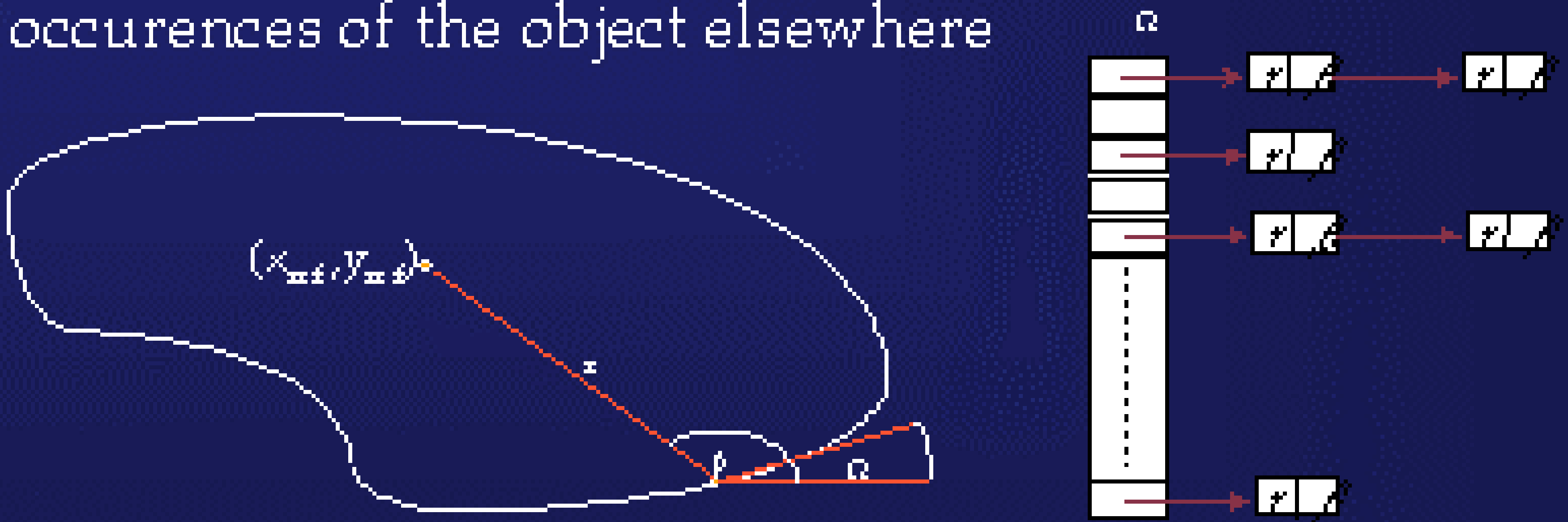
GHT: Building the R-Table

- Using Ω as an index into the R-table we place a tuple storing r and β at the indexed position



GHT: Building the R-Table

- Having done this for each feature point the R-table will fully represent the template object. Also, since the generation phase is invertible we may use it to localise occurences of the object elsewhere

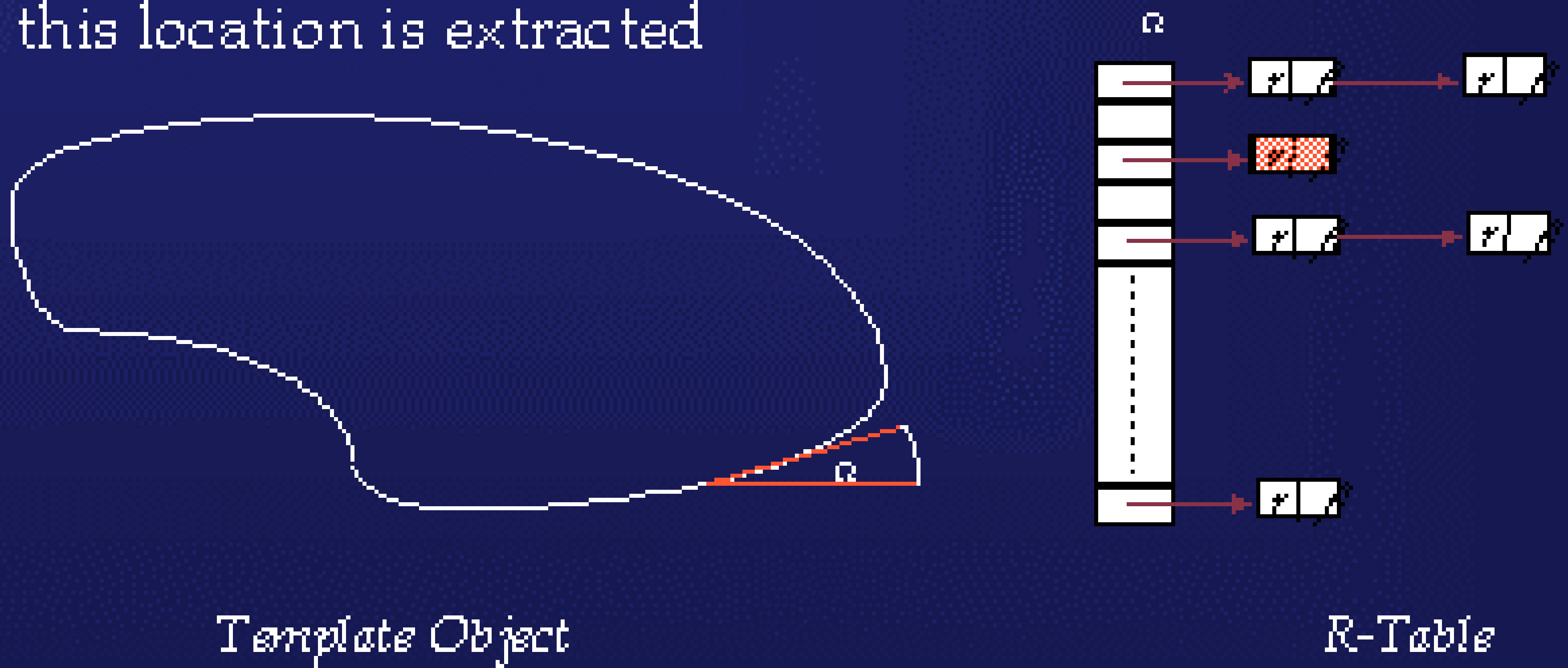


Template Object

R-Table

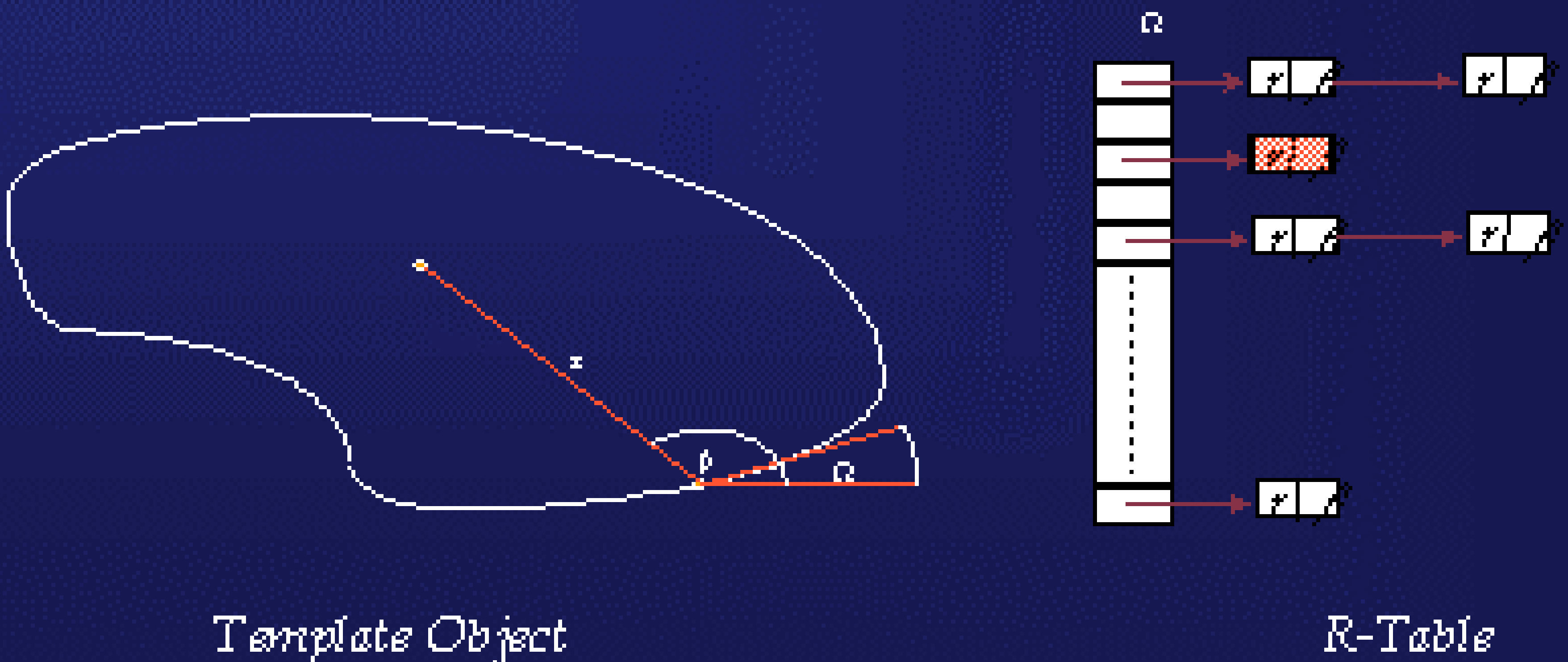
GHT: Object Localization

- Given an unknown image each edge point is segmented and its orientation, Ω, is calculated. Again, using Ω as an index into the R-table each (r,β) pair at this location is extracted



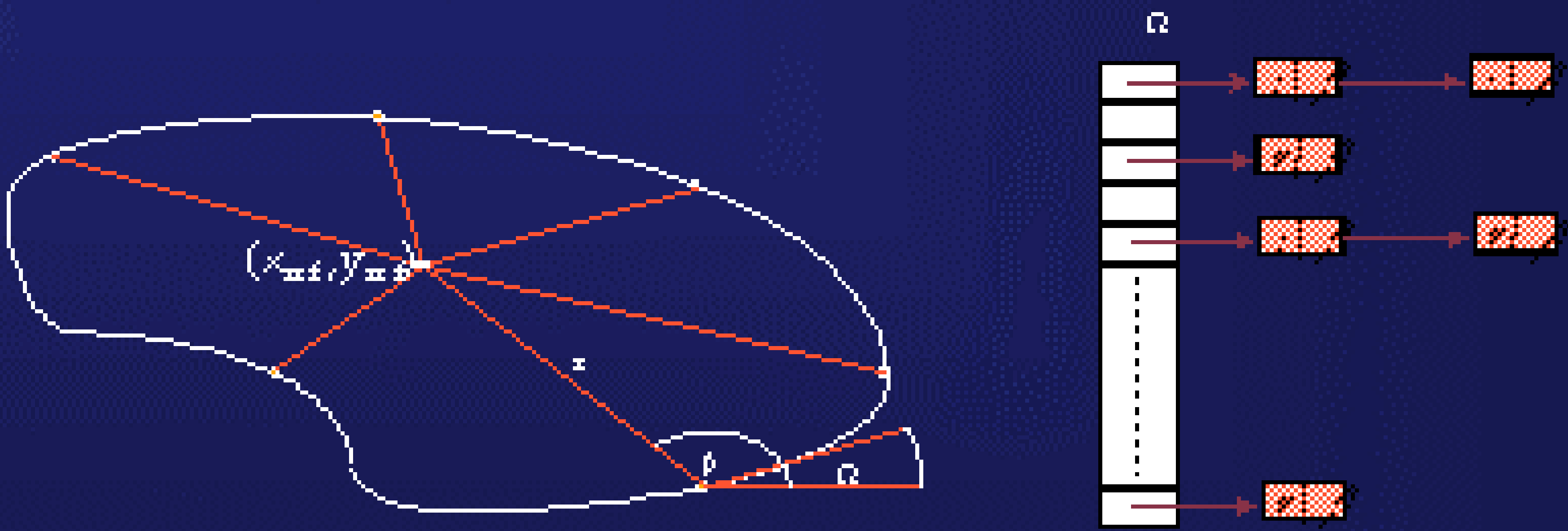
GHT: Object Localization

- Using these (r,β) pairs we calculate each possible position for the reference point.



GHT: Object Localization

- Although some bogus reference points may be calculated, given that the object exists in the image, a maximum will occur at the reference point.



Template Object

R-Table

Conclusions on GHT

- Standard Techniques allow for **<u>invariance</u>** to scale and rotation in the plane

- In general, objects in the real world are 3-dimensional

- Hence a single silhuette provides no invariance to pose (i.e. rotation out of the plane).

- No pose estimation.

- Roll on the SNHT....!

The Surface Normal Hough Transform

The Surface Normal Hough Transform

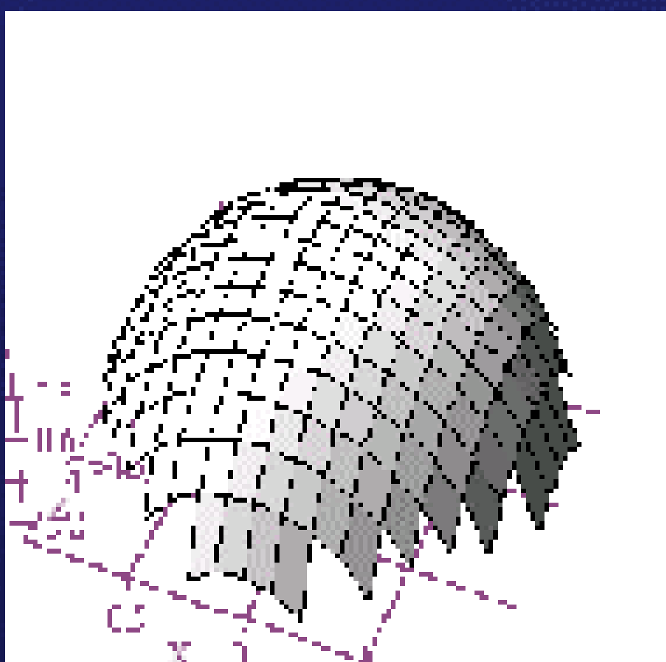
- A technique for computing the *3-D position of a surface* having a specific pose

- A technique for computing the **3-D position and orientation (pose)** of a surface with respect to the pose of a prototypical exemplar of that surface

- Hence a technique for directly registering two **similar surfaces**

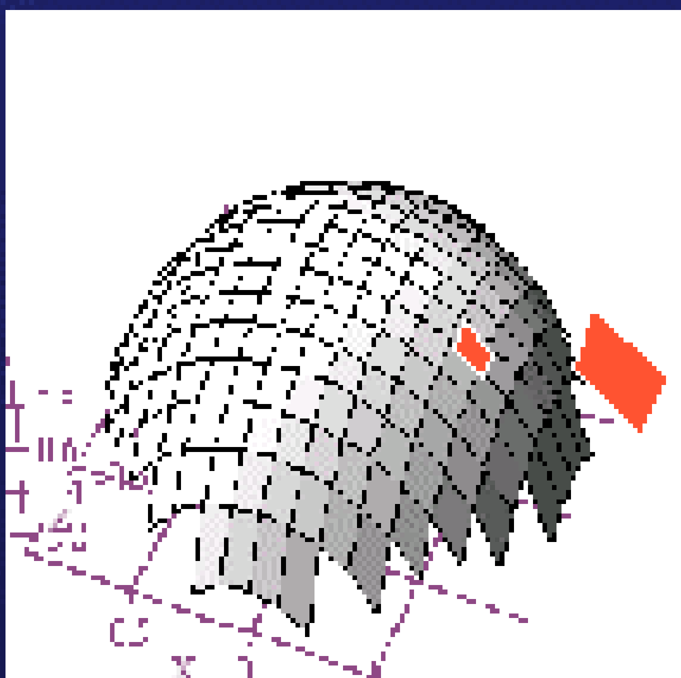
SNHT: Calculating Orientation

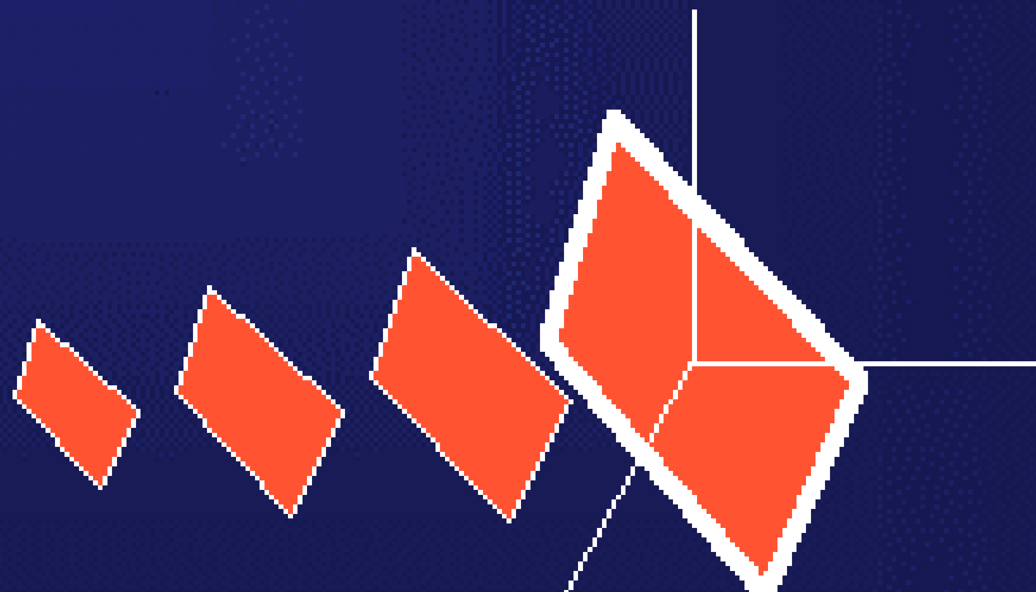
- Suppose we wish to register the surface below with a similar surface



SNHT: Calculating Orientation

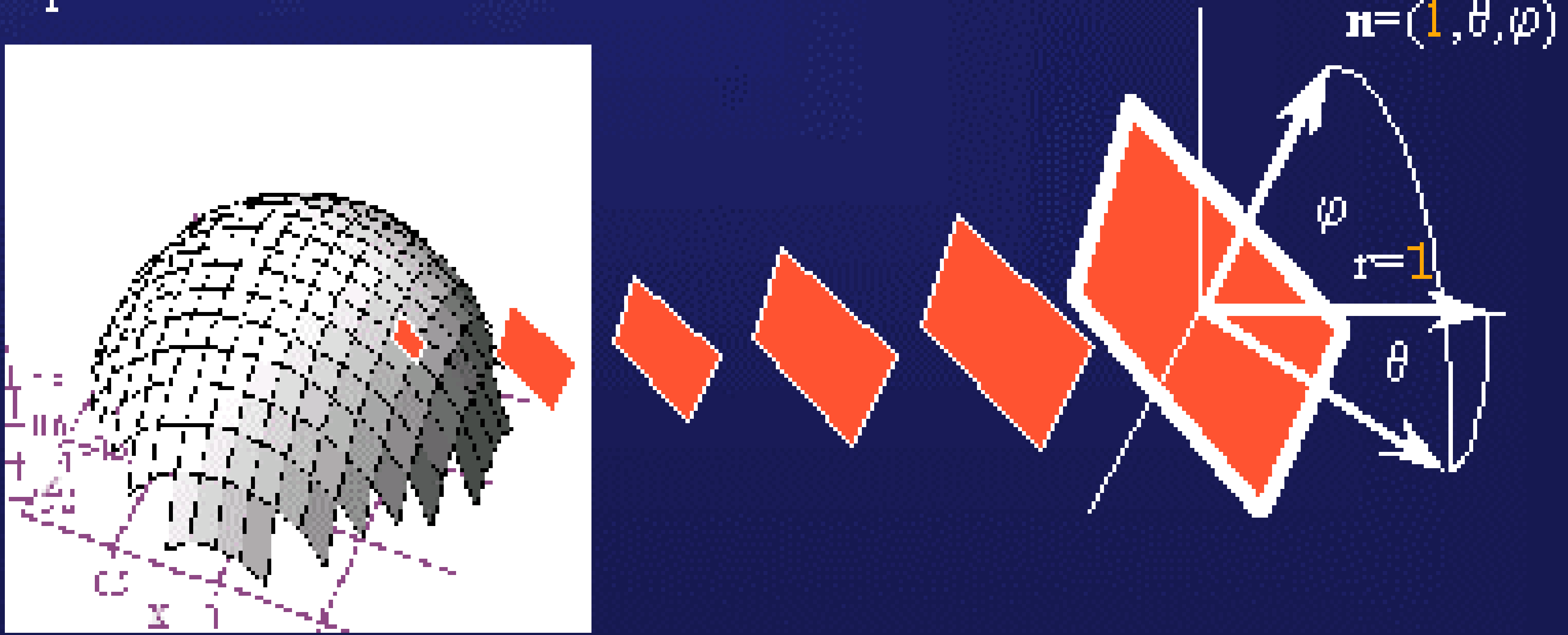
- To use the Hough Transform we must again talk about the orietation at each point





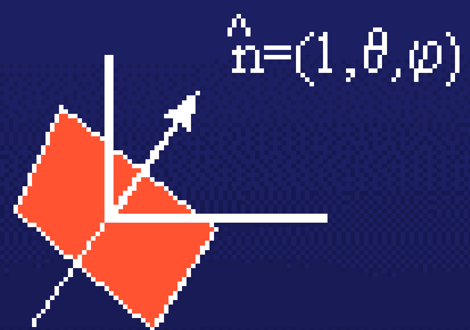
SNHT: Calculating Orientation

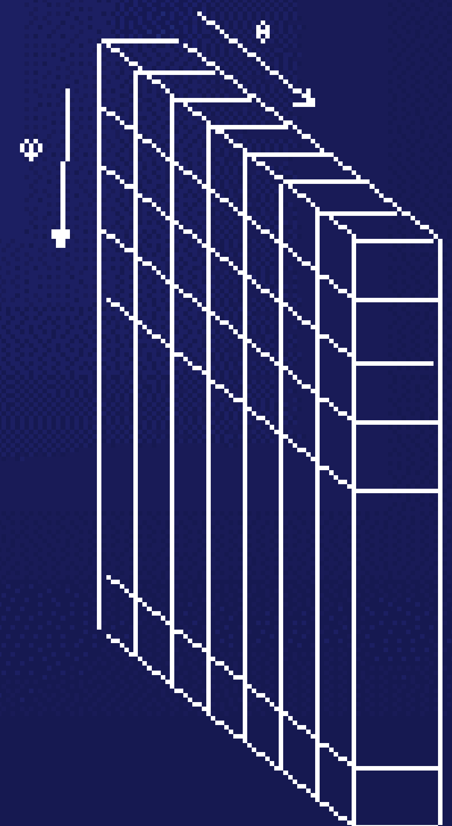
- To specify the orientation at a surface point we use the unit normal vector to the tangential *plane* at that point



SNHT: Building the R-Table

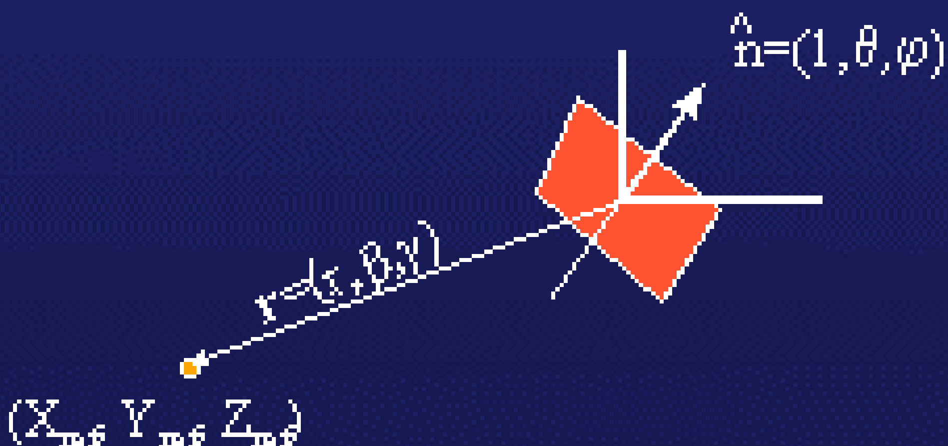
- Specifying the orientation of the unit normal vector requires two variables, hence the R-table becomes a 2-dimensional array of linked lists

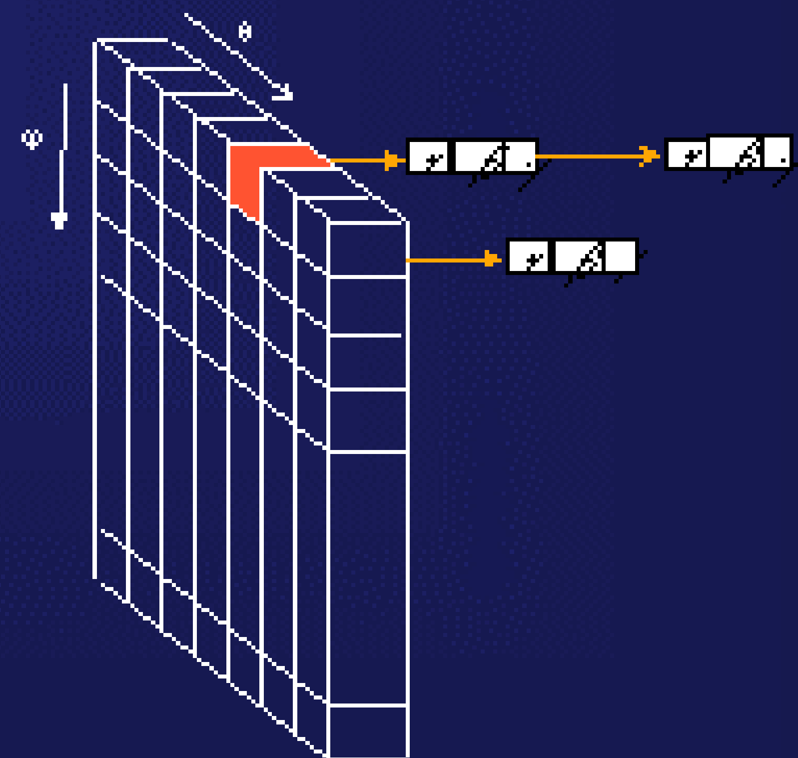




SNHT: Building the R-Table

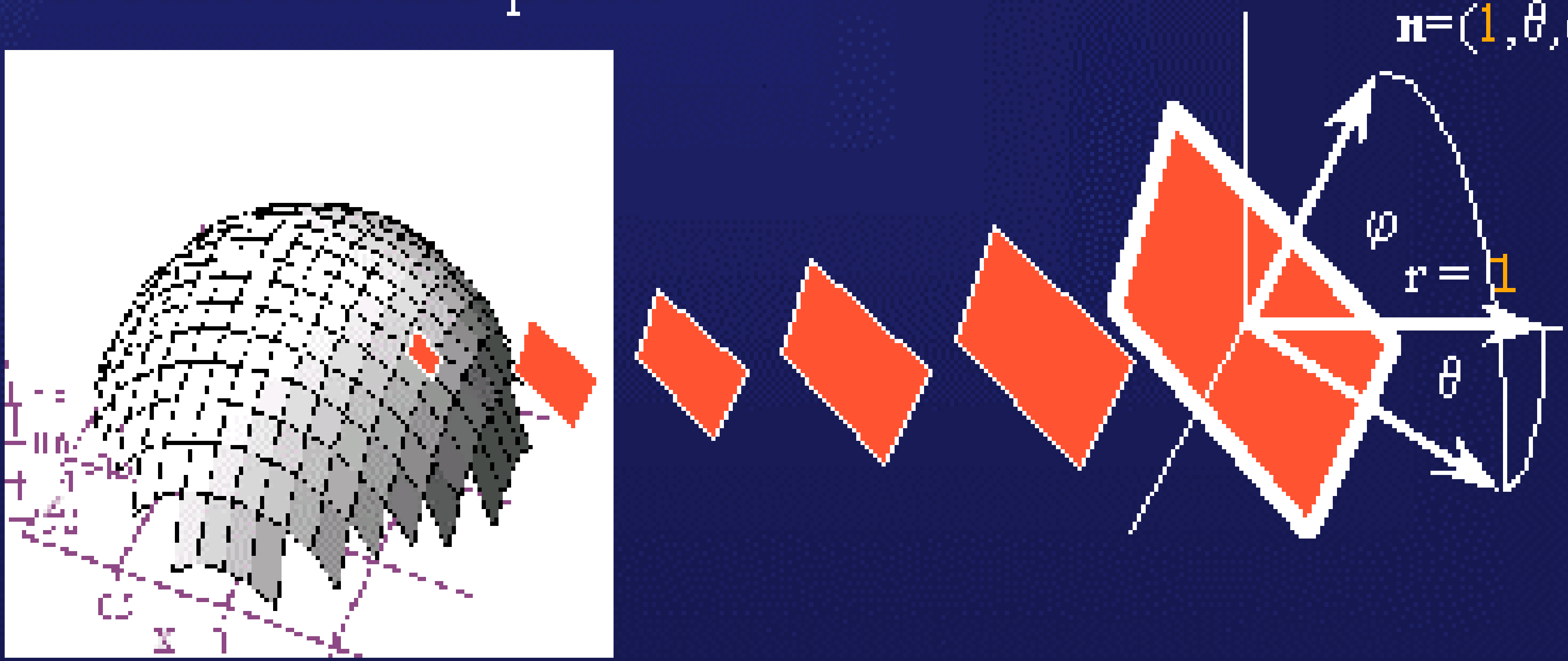
- Indexing in at this location in the R-table we now place a triple (r, β, γ) representing the length and direction of the vector joining the surface point to a chosen reference point





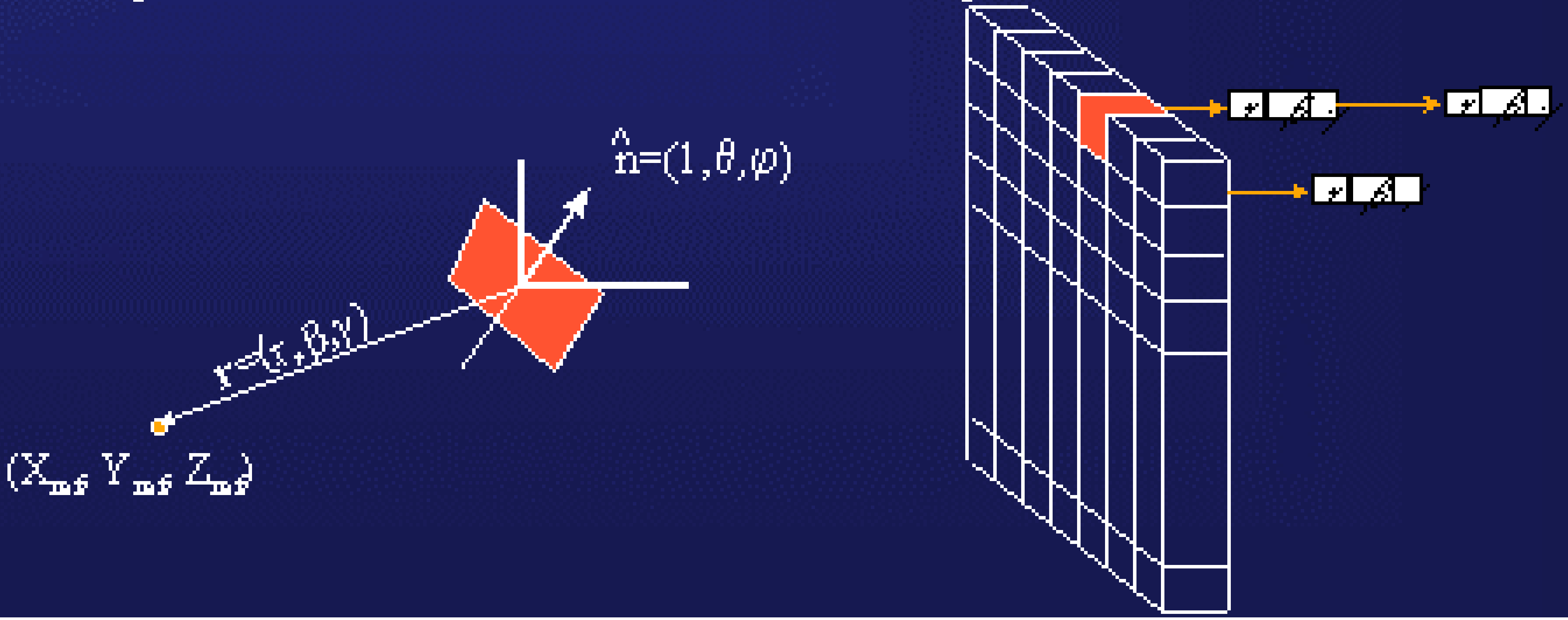
SNHT: Localizing the Object

- Given an surface which we wish to register with the template surface we calculate the unit surface normal at each surface point



SNHT: Localizing the Object

- Using the orientation of this surface normal as an index we extract the (r,β,γ) triples at this location in the R-table and, subsequently calculate the possible positions of the surface reference point



Conclusions and Research Directions

- Technique for registration of surfaces
- Invariant to size and orientation in 3-D
- Can be extended to localization in 2-D
- Implementation
- Validation of simple surfaces
- Face Database for testing of human head pose estimation

Performance Issues

Advantages:

- Works for broken curves
- Uses gradient for speed and further noise removal
- Robust to noise
- Can be extended to a General Hough Transform

Disadvantages

- Expensive when number of parameters is large
- Gradient information can have errors

Gradient Information

- Edge gradient in image space can be used in Hough Transform to reduce one dimension in incrementing the accumulator array
- For line detection the gradient is @, and so need only to vote for one cell (p,@) where p is
  - $p = x_i \cos @ + y_i \sin @$
- For circle detection the gradient is @, and so need only to vote along a line given by the equations
  - $a=x + r \cos @$, $b = y + r \sin @$

Sources

- John Mc. Donald
- D.Vernon