## Navigation: Determining Angle of Chip wrt XY Motor

### User Manual

* Place the tip at the center of a crosshair
* Using the XY Motor, move the tip to the center of an adjacent crosshair
  + Record (Δx, Δy) values and input them into LabView

### Software

* Compute
* Using Fast mode, we can move 3025 μm or 1255 μm to get within the region of an adjacent crosshair
* Input vectors like (±1255, 0) or (0, ±3025) and apply the rotation matrix to convert the vectors from chip to XY Motor coordinates
  + M = [

]

* + - Depending if the chip is right-side-up or up-side-down
* The value computed is just a rough estimate
  + Thus we should do a validation check to see if it is sufficiently accurate
  + This validation check may involve moving in all directions to see if we can get within the region of an adjacent crosshair
    - If not, then have the user redo the process of finding (Δx, Δy) values
  + We can add a bit of flexibility to the validation of value by creating a Check\_Surroundings() function which sends fixed commands to the XY Motor so that we sweep the surrounding area in all directions to try to capture the crosshairs

## Navigation: Fine Adjustment

* I think we need to define a Find\_Tip() method which takes in the Nanocam image and outputs the pixel coordinates corresponding to the tip of the cantilever
  + This is used alongside Find\_Crosshar\_Center() to obtain the difference in row/column of the image and from there adjust the tip so that it’s in Standard Position
  + This also implies that we know how to convert vectors in pixel coordinates to a vector in the XY Motor
* Assuming we can do all of the above so that after getting the tip within the region of a crosshair, we can adjust the tip so that it’s in Standard Position
  + From there, we can have fixed commands to scan at Spot A and Spot B