Capability calculations for 2D and 3D elements

Overview

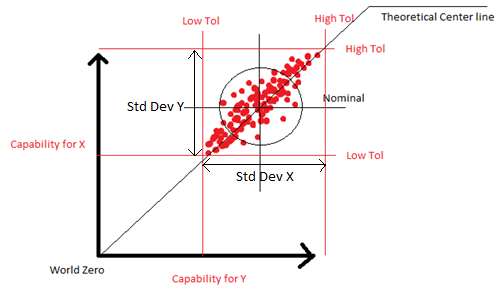
For several years I worked with measurements and capability studies for a large production industry within the auto motion industry, gathering data and analyzing the quality situation from a geometrical standpoint. But something always bothered me, and that was the way we worked out capability for two dimensional and three dimensional requirements. I will try and explain how I look at the issue and come up with a solution that feels more *“true”* with the requirements. This paper will only focus on true position tolerances, how it relates to the theoretical centerline and how you should calculate your capability using this theoretical centerline.

Problem description

The main problem that I want to describe, is the per axis evaluation that most company’s preform to predict quality. This works well for one dimensional requirements but for true position this will not give you an accurate prediction of your capability.

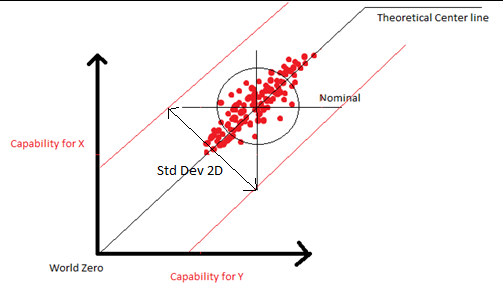
So today we calculate capability for a three dimensional object by looking at the x, y and z values separately. So for a true position tolerance we look at; capability for x, capability for y and capability for z. And if one axis fails, we deem the dimension to not being capable. It is also important to mention that the measurements are in relation to the worlds zero point and have not been transformed to the relation of the requirement (centerline).

**If we look at illustration 1:**



*In this case, the elements theoretical centerline is at a 45 degree angle to the world.*

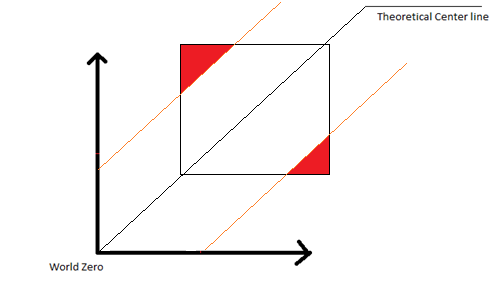
In this illustration you can see that the standard deviation will be high for both x and y. But we are not measuring in relation to the theoretical centerline. And if we have a true position requirement, it states: - *Closest distance from the point to the theoretical centerline* making this evaluation false. The illustration also shows that the shape of Low Tol and High Tol have no relation to the actual tolerance area of the element.



This illustration shows the tolerance in the appropriate way in relation to the requirement. And here the standard deviation will me much lower. It does not only make the standard deviation smaller, it also make the shape of the tolerance accurate to the true position requirement.

If you think in three dimensions, the true position tolerance should be represented by an infinite cylinder aligned with the theoretical centerline, not as in illustration 1 where it is a Box.

One of the biggest issues I have with the “box” model, is that you are using tolerance areas that you don’t have (see illustration 3).



If we compare the 2 tolerance areas, the box shapes extends beyond the limits set by the requirements, giving you false data about the outcome of your capability study.

*I can somewhat understand the use of per axis capability studies when it comes to CNC machines where it gives you a good indication of with axis preforms the best and if there is an need for maintenance.*

*What do we want?*

*So what is it that we want to achieve with all of this? We want to know the likelihood of our measurements being within the set tolerance.*

Calculating Standard Deviation

So how should we preform the calculations? First we need to establish what you need for a standard deviation calculation in 2 or 3 dimensions.

First we need a mean. If you look at the box model, it is the mean of X and the mean of Y but separate. We need to establish a 2 dimensional mean and we will call it m(x, y); And this is straight forward, just add the mean from x and y into m, m(mean of x, mean of y);

We now need to shift the centerline from nominal to the mean, so that we can correctly calculate the deviation for each element.

We calculate the new position px, py and pz and use dx, dy and dz for the theoretical centerline and x, y, and z are the position of the element.

***Vector Subtraction***

px = x - mx

py = y - my

pz = z - mz

***Cross product***

cx = (py \* dz) - (pz \* dy)

cy = (pz \* dx) - (px \* dz)

cz = (px \* dy) - (py \* dx)

***Vector Distance***

True Position = Sqr((cx \* cx) + (cy \* cy) + (cz \* cz))

Now we have the true position of the element to the mean in relation to the theoretical centerline. Next we add all true positions for the elements and divide it by the number of elements, and that is our Standard deviation.

Sum(Sqr((x.cx \* x.cx) + (x.cy \* x.cy) + (x.cz \* x.cz))) / numElements

Calculating Tolerance

So to calculate Cpk we now have established the standard deviation. And to be able to conclude our calculation for (Tol – Mean) / (StdDev\*3) we need to have the accurate relation between Tol and Mean.

So for the true position you state within what dimension the element must be. So for a true position of 6, the tolerance is +/- 3. So we need to know the distance of the mean to the center line witch we do by: SqareRot((mean.x \* mean.x)+(mean.y \* mean.y)+(mean.z + mean.z)).

Adding this mean to the Tol – Mean will give us the space between the mean and the tolerance that the 3sigma needs to fit. And remember that we need to divide the true position tolerance by 2 to get a distance. So Area = (Tolerance / 2) – Mean Distance.

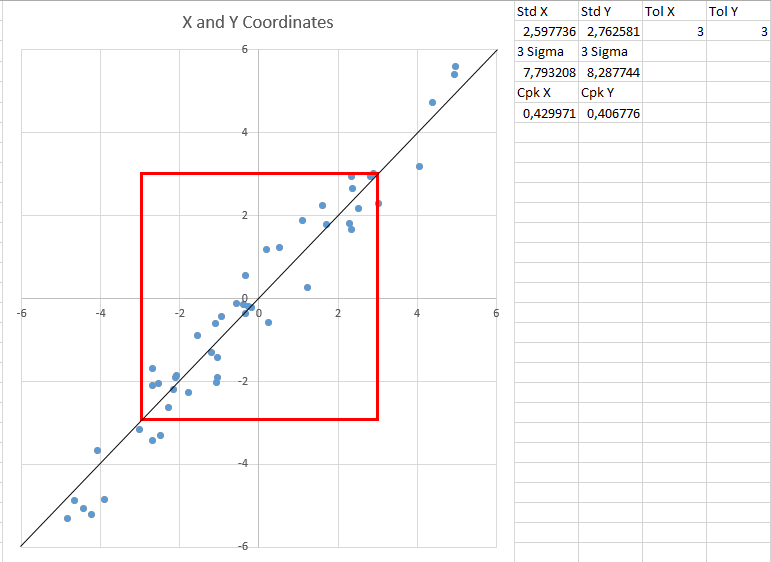
Final Calcualtion

Now we have all the parameters to do our Cpk calculation:

Cpk = Area / 3StdDeviation

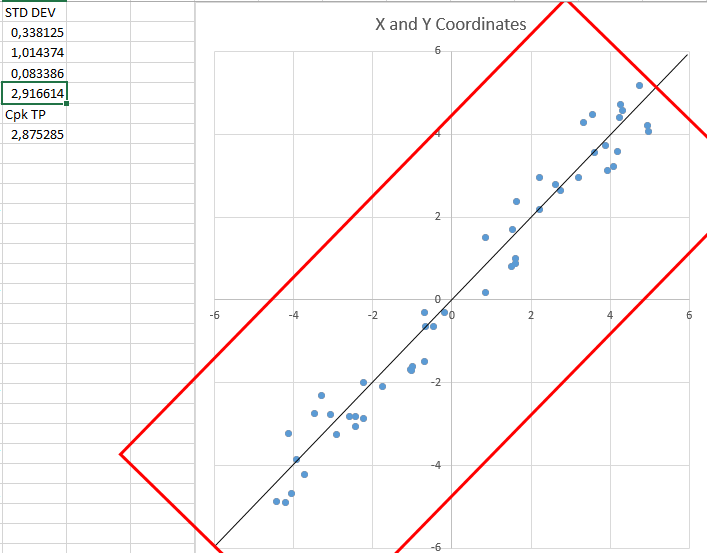
I have made a chart to demonstrate the difference of doing this per axis and doing this as a multidimensional calculation.

Per axis:



I only did this for an upper tolerance since I know that it will be basically the same if we do upper and lower tolerances. Here I have marked the tolerance with a 3x3 rectangle to illustrate the tolerance that is being used when calculating Cpk per axis. The standard deviations are very high and will never be accepted for capability.

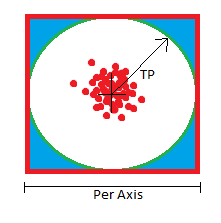
If we instead look at the multidimensional chart:



The tolerance is the same, but we have aligned it more properly and you can see that the standard deviation is much smaller. This gives us more “*true”* data for what the problem for the measurements are.

Other Issues

The other issue that per axis Cpk brings is the shape of the tolerance when you look through the centerline. As you can see in the illustration, the calculation area for per axis is exceeding the actual tolerance.



Conclusion

There are various issues when not preforming multidimensional calculations for multidimensional requirements. The most obvious one for me is that you are using tolerances areas that you don’t actually have. But you bay also find it more complicated to get a good Cpk value since the production equipment and product engineers have designed everything to meet the specified requirements and those are not single axis requirements.

I am also surprised that various statistical software’s do not support three dimensional capability. I really understand the benefits of studying measurements per axis, but you cannot blindly ignore the shape of the requirement.